BUY IN PRINT WORLDWIDE MAGPI.CC/STORE

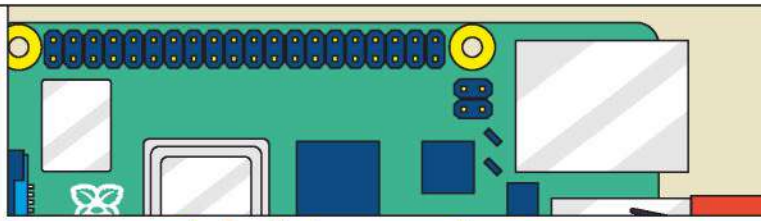
The MagPi

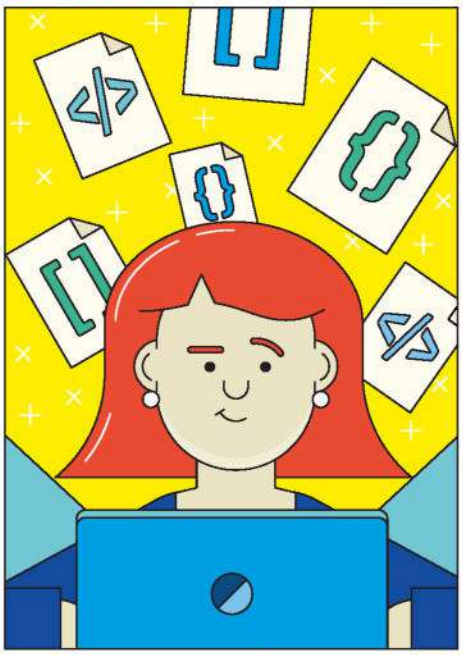
Issue 128 | April 2023 | magpi.cc

The official Raspberry Pi magazine

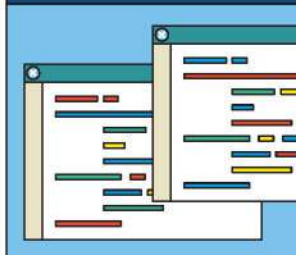
LEARN TO CODE WITH PYTHON

Discover coding with Raspberry Pi

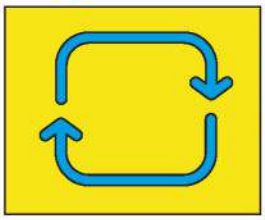


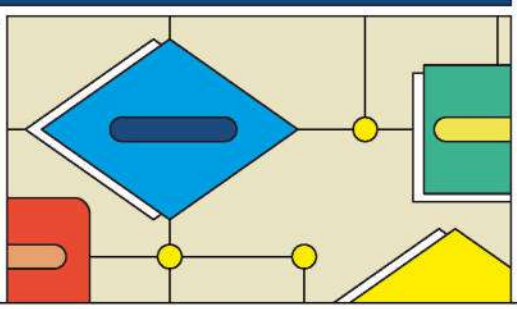


TAKE NIGHT SKY PHOTOS



GET STARTED WITH PICO CIRCUITS



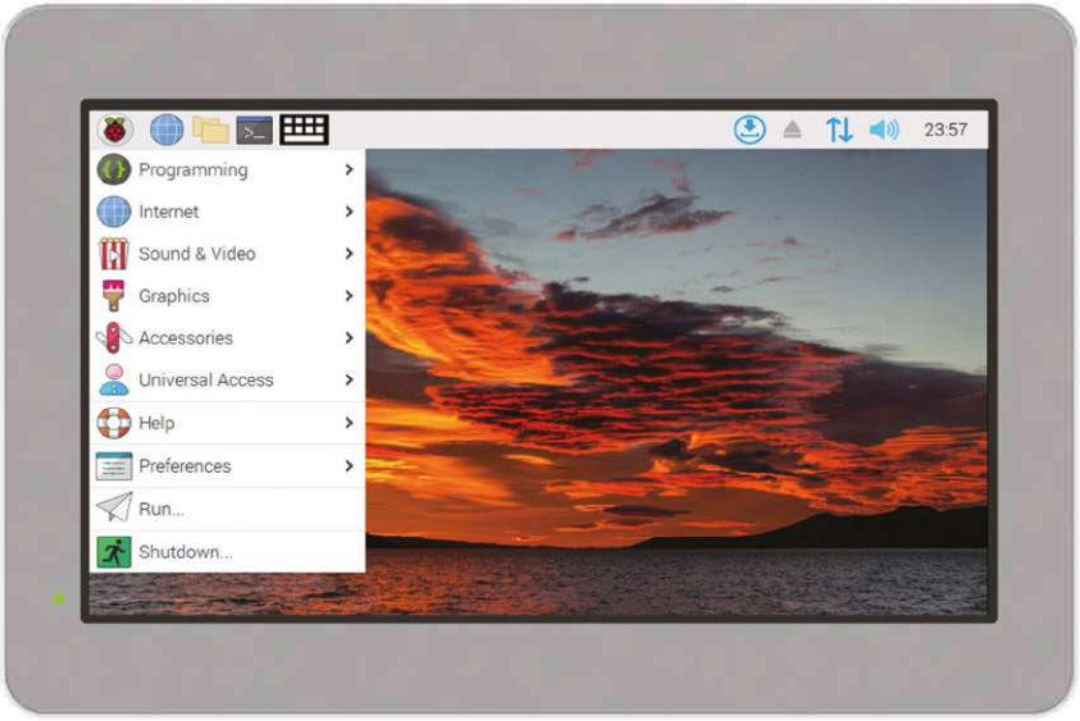


GLOBAL DELIVERY
magpi.cc/store
Raspberry Pi PRESS

NEW! RASPBERRY PI GLOBAL SHUTTER CAMERA

UL LISTED

Industrial Raspberry Pi

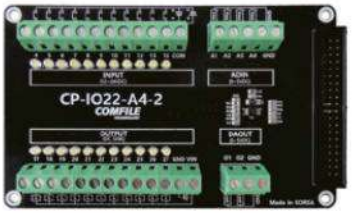


ComfilePi

The ComfilePi is a touch panel PC designed with high-tolerant components and no moving parts for industrial applications. It features a water-resistant front panel, touchscreen, color LCD (available in various sizes), RS-232, RS-485, Ethernet, USB, I2C, SPI, digital IO, battery-backed RTC (real-time clock), and piezo buzzer.

Use the rear-panel 40-pin GPIO header to expand its features and capabilities with additional I/O boards. The ComfilePi is UL Listed and employs Raspberry Pi Compute Module.

POWERED BY Raspberry Pi



Visit www.comfiletech.com

© copyright COMFILE Technology, Inc. ALL RIGHTS RESERVED

COMFILE TECHNOLOGY

WELCOME
to The MagPi 128

Learning to code can be a life-changing experience. Computers are useful things, and learning how to get under the hood is – in itself – pretty rewarding, and there's plenty of demand for coders worldwide.

But beyond that, learning to program teaches you how to think. You learn how to break down complex tasks and put things in order. It teaches you how to solve seemingly impossible problems. Our Learn to Code with Python feature (**page 44**) can help you out on this journey. I hope you take it.

You'll also discover excellent Computer Science resources in this issue (**page 80**), including CS50 and MITx (both of which helped me immeasurably when I got serious about coding).

Blending code with hardware is what *The MagPi* is all about. And you should also take a look at Stewart Watkiss' electronics series, which moves into microcontrollers (**page 52**).

Being a coder is great fun. I hope this edition of *The MagPi* helps you get started.

Lucy Hattersley Editor

EDITOR

Lucy Hattersley

Lucy is the editor of *The MagPi* and would like to petition for a Raspberry Pi office puppy. Or she'll build another robot

magpi.cc

GET A RASPBERRY PI PICO W WITH A SUBSCRIPTION! PAGE 42

YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
The official Raspberry Pi magazine
LEARN TO CODE WITH PYTHON
Discover coding with Raspberry Pi
NEW! RASPBERRY PI GLOBAL SHUTTER CAMERA

WIZnet

TOE

TCP/IP Offload Ethernet

Design Contest

- Supporting up to 80 Mbps Speed
- Targeting High Performance Applications

USE W5300

- 8/16 bit Data Bus Interface (Direct & Indirect Address Mode)
- High Network Performance : Max 80Mbps (by DMA)
- 8 Independent hardware sockets

Buy

W5300($5) or W5300 TOE Shield($10)

at WIZnet e-shop

W5300

W5300 TOE Shield

WIN A CHANCE FOR $ 30,000 IN TOTAL

1st PRIZE	2nd PRIZE	3rd PRIZE
$ 5,000 x 2	$ 2,500 x 2	$ 1,000 x 10

Contest begins	Mar 16
Get a Sample	Mar 16 ~ April 30
Submissions close	Jun 30
Winners announced by	July 18

at maker.wiznet.io

Contents

Issue 128 > April 2023

Cover Feature

44 Learn to Code with Python

Regulars

08 World of Raspberry Pi news
40 Case Study: Yoto Player
90 Your Letters
92 Community Events calendar
97 Next Month
98 The Final Word

Project Showcases

14 Heavy Pan Tilt System
18 Word Clock
20 Game Boy Interceptor
24 Aquarium dosing pump
26 Tank Driving Simulator
30 Progress Bar
32 BrachioGraph
34 Herbie_Bot
36 CinePI

Heavy Pan Tilt System

Tank Driving Simulator

The MagPi is published monthly by Raspberry Pi Ltd, Maurice Wilkes Building, St. John's Innovation Park, Cowley Road, Cambridge, CB4 0DS, United Kingdom. Publishers Service Associates, 2406 Reach Road, Williamsport, PA, 17701, is the mailing agent for copies distributed in the US and Canada. Application to mail at Periodicals prices is pending at Williamsport, PA. POSTMASTER: Send address changes to The Magpi, c/o Publishers Service Associates, 2406 Reach Road, Williamsport, PA, 17701.

Tutorials

52 Introduction to microcontrollers
56 Build an ML digital transcriber
60 Make a binary clock

The Big Feature

66 Astrophotography with Raspberry Pi

Reviews

72 Vizy Camera
74 Meteor 10.1" touchscreen
76 Alpakka controller
78 Ten amazing displays
80 Learn computer science

Community

84 Trevor Warren interview
86 This Month in Raspberry Pi

Build an ML digital transcriber

Make a binary clock

Ten amazing displays

This Month in Raspberry Pi

DISCLAIMER: Some of the tools and techniques shown in The MagPi magazine are dangerous unless used with skill, experience, and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of yourself and your equipment. Children should be supervised. Raspberry Pi Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in The MagPi magazine. Laws and regulations covering many of the topics in The MagPi magazine are different between countries, and are always subject to change. You are responsible for understanding the requirements in your jurisdiction and ensuring that you comply with them. Some manufacturers place limits on the use of their hardware which some projects or suggestions in The MagPi magazine may go beyond. It is your responsibility to understand the manufacturer's limits.

SELECTION
STARTS HERE
Digi-Key ELECTRONICS
More products. More name-brand suppliers.
More new product inventory. We simply have more of what you need.
Find it at digikey.co.uk or call 0800 587 0991.
Digi-Key ELECTRONICS
FREE SHIPPING ON ORDERS OVER £33 OR $50 USD*
*A shipping charge of £12.00 will be billed on all orders of less than £33.00. A shipping charge of $18.00 USD will be billed on all orders of less than $50.00 USD. All orders are shipped via UPS, Federal Express, or DHL for delivery within 1-3 days (dependent on final destination). No handling fees. All prices are in British pound sterling or United States dollar. Digi-Key is a franchised distributor for all supplier partners. New products added daily. Digi-Key and Digi-Key Electronics are registered trademarks of Digi-Key Electronics in the U.S. and other countries. © 2023 Digi-Key Electronics, 701 Brooks Ave. South, Thief River Falls, MN 56701, USA
ECIA MEMBER
Supporting The Authorized Channel

Global Shutter Camera

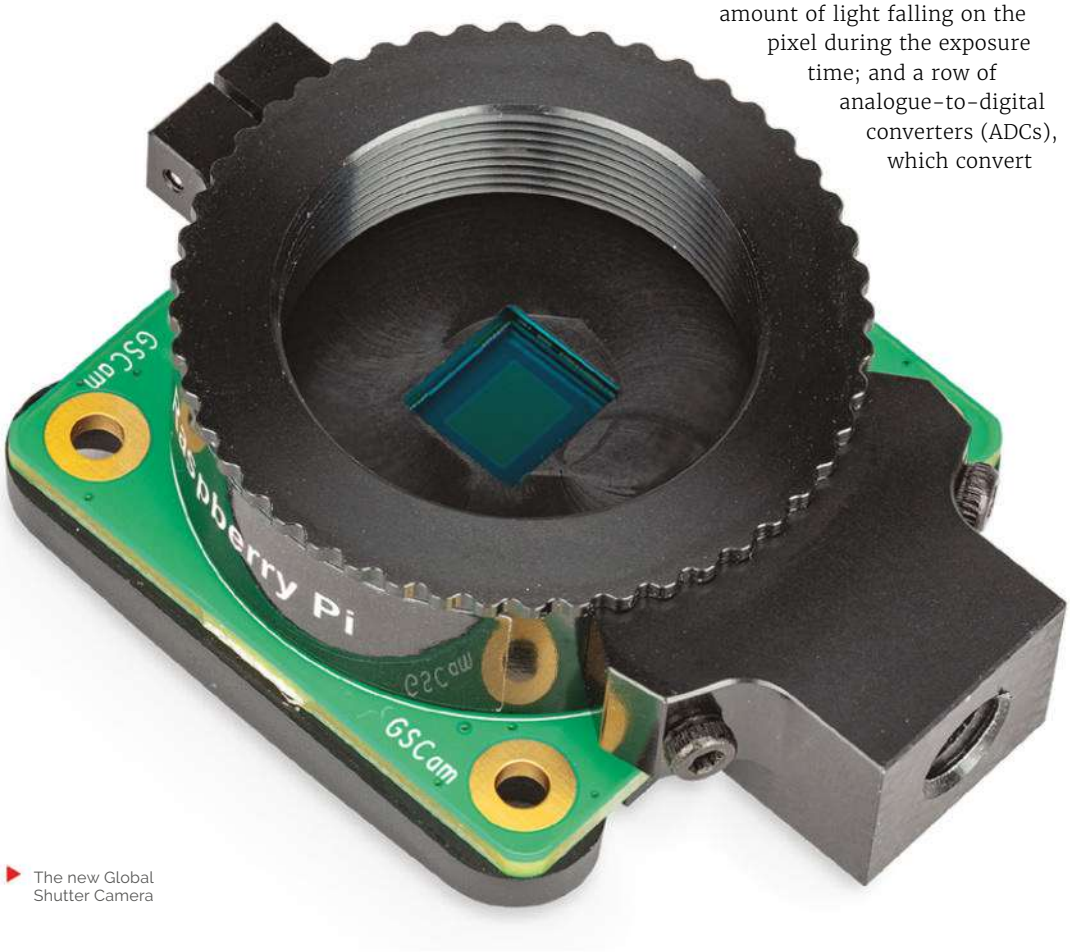
New Raspberry Pi Global Shutter Camera for machine vision and more. By **Eben Upton**

Remember those new cameras we launched way back in the dawn of… January? [see **magpi.cc/126** – Ed]. We have another one for you – and this time we're getting specialised. We're launching something a little different: the Raspberry Pi Global Shutter Camera, available now at $50 (**magpi.cc/globalshuttercam**).

Built around Sony's 1.6-megapixel IMX296 sensor, the Global Shutter Camera is able to capture rapid motion without introducing rolling shutter artefacts. This makes it a great fit for sports photography and machine vision applications, where even small amounts of distortion can seriously degrade inference performance.

Rolling shutters, global shutters

Every camera we've released to date, from 2014's Camera Module 1 to our High Quality Camera and beyond, has used a rolling shutter sensor. These sensors have a two-dimensional array of light-sensitive pixels, which generate an analogue value proportional to the amount of light falling on the pixel during the exposure time; and a row of analogue-to-digital converters (ADCs), which convert



The new Global Shutter Camera

The Global Shutter Camera combines the C/CS-mount of the High Quality Camera with the Sony IMX296 sensor

these analogue values into digital values which are fed back to your Raspberry Pi.

The row of ADCs is connected to each row of the pixel array in turn, so each row is sampled at a slightly different time. Provided there is no motion in the scene, this isn't a problem, but once things start to move – and particularly if something is moving fast – we start to see rolling shutter artefacts. Linear motion results in compression, stretching, or shearing of the moving object. Rotary motion can create some very odd-looking shapes indeed. Severe rolling shutter artefacts are unsightly, and hard to detect and correct, but even imperceptible artefacts can interfere with the operation of machine vision algorithms. If we want to eliminate them altogether, we need to use a global shutter sensor. This pairs each pixel with an analogue storage element; when the shutter fires, each pixel immediately copies its analogue value into its storage element, from where it can be read and converted at leisure.

> When the shutter fires, each pixel immediately copies its analogue value into its storage element

The storage element adds complexity and area to each pixel. Global shutter sensors tend to have a lower resolution than rolling shutter sensors of the same size: contrast the 7.9mm, 12-megapixel IMX477 sensor used in the High Quality Camera with the 6.3mm, 1.6-megapixel IMX296.

Credits

Like all recent camera products, the Global Shutter Camera hardware was designed by Simon Martin. Naush Patuck, Nick Hollinghurst, David Plowman, Serge Schneider, and Dave Stevenson wrote the software. Alasdair Allan, Simon Martin, David Plowman, Andrew Scheller, and Liz Upton worked on documentation. Austin Su led on sourcing. Jack Willis designed the packaging, and Brian O Halloran (not included with camera) took the photos and video. We'd like to acknowledge the assistance of Phil Holden and John Conroy at Sony, and of Shenzhen O-HN Optoelectronic.

If you think my front is cool… there's a surprise logo for the rear view fans

Enter the Raspberry Pi Global Shutter Camera

Despite the challenges associated with rolling shutter artefacts, our existing cameras are widely used in hobbyist and industrial machine vision applications. And we've seen some real ingenuity: to compensate for artefacts when imaging products on a high-speed conveyor belt, one of our industrial customers ended up training their models using pre-sheared input data.

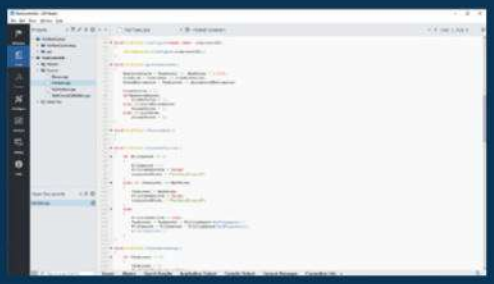
For these applications, a global shutter sensor offers clear advantages. And reduced resolution isn't a problem, as high-resolution images are generally down-sampled before being fed into machine vision models.

The Raspberry Pi Global Shutter Camera combines the C/CS-mount metalwork of our High Quality Camera with Sony's IMX296 sensor. It is compatible with the same broad variety of lenses, including the 6 mm CS-mount and 16 mm C-mount CGL lenses that we offer through our Approved Reseller partners.

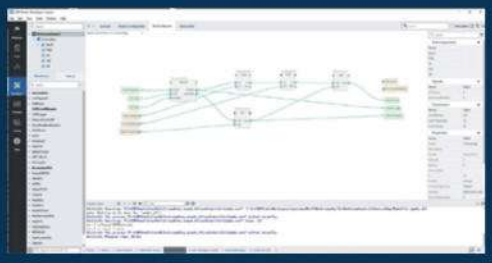
Like all our camera products, you can use the Global Shutter Camera with any Raspberry Pi computer that has a CSI camera connector, and we've updated our hardware documentation (**magpi.cc/cameradocs**) to include everything you need to know about the new product. You'll need to update Raspberry Pi OS to use the new camera: `sudo apt update; sudo apt full-upgrade; sudo reboot` and you're good to go.

Camera in action

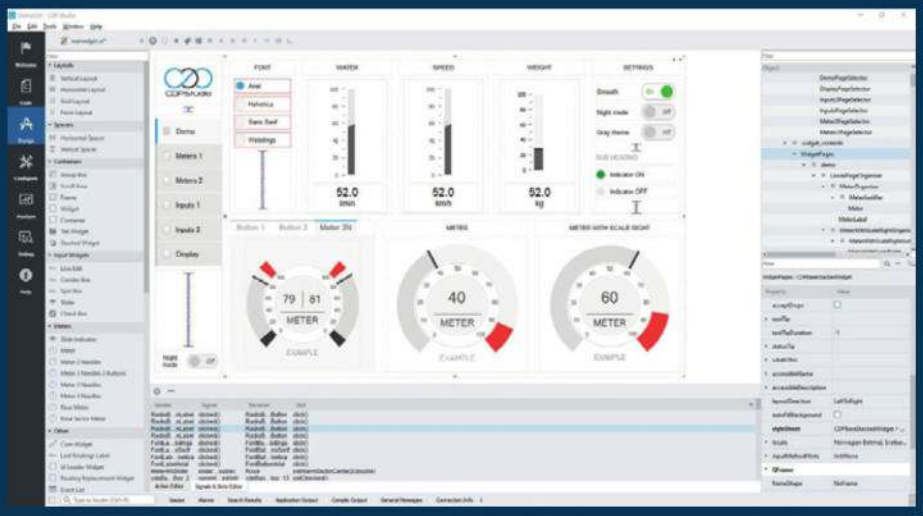
The video at **magpi.cc/gscamaction** illustrates the benefits of a global shutter in the presence of rapid rotary motion. First, we see the output from the High Quality Camera, showing distinctive rolling shutter artefacts, and then we see the artefact-free output from the Global Shutter Camera.



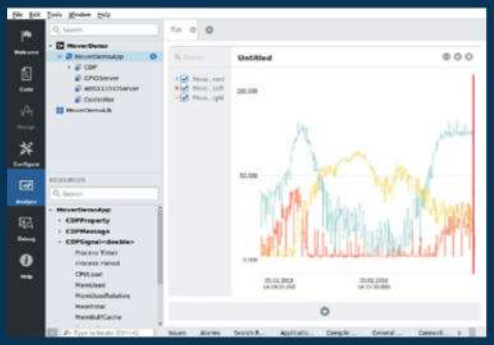
Full-Code



No-Code / Configure



Design



Analyze

now with a

No-Code WEB UI Designer

PROFESSIONAL CONTROL SYSTEM DEVELOPMENT TOOL

Home projects made easy.

CDP Studio, a great software development tool for your home projects. Build systems for Raspberry Pi, use C++ or NoCode programming, open source libraries, out of the box support for GPIO, I2C, MQTT, OPC UA and more. Create beautiful user interfaces. Built for industrial control system development, **FREE for home projects**.

cdpstudio.com
Tel: +47 990 80 900 • info@cdptech.com
CDP Technologies AS // Hundsværgata 8, 6008 Ålesund, Norway

CDPStudio

Raspberry Pi Pico Windows Installer

A simpler solution to getting Pico up and running with Windows computers.

By **Gordon Hollingworth, Raspberry Pi, Director of Software Engineering**

Raspberry Pi is introducing Raspberry Pi Pico Windows Installer, a simple solution to install everything you need to develop for Raspberry Pi Pico, and for other RP2040-based boards, using C or C++ on Windows.

The Raspberry Pi Pico C SDK (**magpi.cc/picocsdk**) can initially seem quite daunting. There are a lot of moving parts to install before you can blink your first LED on, and then off again, especially if you want proper debugging using something like our new Debug Probe (**magpi.cc/debugprobe**). While there has been a 'one-click' solution (**magpi.cc/picosetup**) to install the toolchain on a Raspberry Pi since launch – see Chapter 1 of the Getting Started guide (**magpi.cc/gettingstartedpico**) – installing the toolchain on other platforms, like macOS or Windows, has always been a little more difficult.

When we launched Raspberry Pi Pico and its SDK, our assumption was that people would be happy with a Linux-based toolchain. However, we also documented how to get the toolchain up and working on both Apple's macOS and Microsoft Windows for folks that wanted to use those platforms. Unfortunately, while installing the SDK on macOS isn't too different from doing so under Linux, and works almost out of the box, installing things on Windows is much more difficult; it's just a very different environment.

Fortunately, an enterprising young engineer called Nikhil Dabas (**magpi.cc/ndabas**) decided to see whether he could do better. It turned out, of course, that he could, so I asked him to help us create something even better: a complete integrated installation package that would give people a nice, easy way of building and debugging the Pico SDK.

▼ The new Raspberry Pi Debug Probe

Installing the toolchain

Installing the toolchain is now as simple as downloading, and running, the Pico Installer (download link: **magpi.cc/picosetupwindows**). At the end, the installer will offer to clone and build the Pico examples (**magpi.cc/picoexamples**), giving you the option of changing where the examples are installed. Leave that ticked, and the installer will open a command window to do the initial checkout and build of all the Pico repositories. Once done, you can safely close this window.

Starting Visual Studio Code

In your Start Menu, look for the 'Pico – Visual Studio Code' shortcut, in the 'Raspberry Pi' folder. The shortcut sets the required environment variables and then launches Visual Studio Code.

The first time Visual Studio Code is launched using the Start Menu shortcut, it will open the **pico-examples** folder. To open the folder later, use the 'Open Recent' or 'Open Folder' menu options and navigate to your **Documents/Pico-v1.5.0/** directory, or wherever you installed the examples.

Building the Hello Serial example

If you previously didn't have VS Code installed, everything should 'just work' after installation. But often we've seen problems due to various random settings inserted by other extensions, or by the user, in an existing installation. If this is the case for you, please go to the Pico Installer Wiki (**magpi.cc/picowindowswiki**) for a checklist of known issues and solutions.

Visual Studio Code may ask if you want to configure the pico-examples project when it is first opened; click 'Yes' on that prompt to proceed, if you miss the prompt look for the 'bell icon' in the bottom right. If it doesn't ask, you can click on the blue status bar where it says 'No active kit' and select Pico ARM GCC.

To build the example, click the CMake button on the sidebar. You should be presented with a tree view of the example projects; expand the 'hello_world' and 'serial' trees, and click the small 'build' icon to build that specific project. You should take a note of the [hello_serial] tag down in the CMake toolbar at the bottom of the screen, you can use this to change the target at any time.

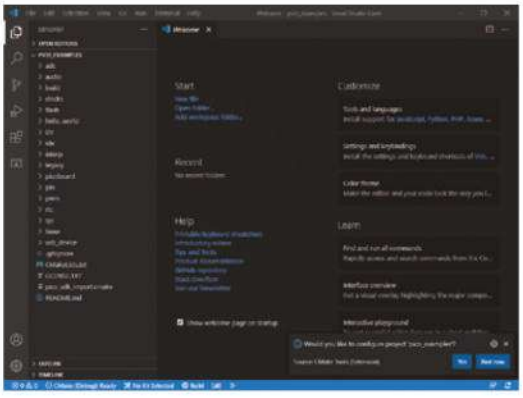
> A simple solution to install everything you need to develop for Raspberry Pi Pico

Debugging your example application

To debug your hello_serial example, you should use the new Raspberry Pi Debug Probe (**magpi.cc/debugprobe**) or set up your own PicoProbe using a second Raspberry Pi Pico. If you don't have a Debug Probe, see Appendix A of the Getting Started (**magpi.cc/gettingstartedpico**) guide for instructions on setting up a second Pico as a PicoProbe.

To quickly check you've set this up properly, in Visual Studio Code, click on the SERIAL MONITOR tab. When you plug in your Debug Probe (or PicoProbe) you should see an additional COM port in the port drop-down list. You should leave the baud rate set to the default of 115,200 in most cases.

Click 'Start Monitoring' to open the serial port.



The pico-examples repository open in Visual Studio Code

Raspberry Pi Debug Probe connected to a Dell computer running Windows

Let's start debugging

Assuming you have wired up the Debug Probe to your Pico correctly, press **F5** to start debugging or click the Run and Debug button (**CTRL+SHIFT+D**) on the sidebar and then the small 'play' icon at the top of the debug window.

Your selected target should now be built, uploaded, and started. The debugger interface will load and will pause the execution of the code at the main() entry point. At this point, you can use the usual debugging tools to step, set breakpoints, inspect memory, and so on. Hit 'Run', and you should now be able to switch back to the SERIAL MONITOR tab to see the serial output appear.

Next steps

For more information on creating your own project outside of the pico-examples, see the tutorial at **magpi.cc/piconewproject**, or read Chapter 8 of the Getting Started guide (**magpi.cc/gettingstartedpico**) for a full walkthrough.

Heavy Pan Tilt System

Keen photographer Vigasan needed a solid stand for his builds, and found Raspberry Pi ideal, he tells **Rosie Hattersley**

MAKER

Vigasan

With 20 years in the medical field as an electronic engineer, Vigasan frequently uses Raspberry Pi for both work and his hobby projects.

magpi.cc/motorizedcamgit

Electronics engineer Vito counts astronomy and photography among his interests. After using Raspberry Pi a few years ago as part of his role in the medical sphere, he realised it would fit rather neatly with his hobbies too. This led to him setting up the YouTube channel Vigasan (**magpi.cc/vigasanyt**) to showcase his projects.

The Heavy Pan Tilt System addresses a common issue with astrophotography: the need for a setup robust enough to take a hefty digital SLR camera and have it pan smoothly across the night sky without juddering or losing either focus or the object in the viewfinder. Commercial options tend to be "quite expensive astronomical mounts," so Vito set about create his own version using his own mechanical designs and software to automate object tracking. Raspberry Pi provided the critical stepper motor controls and OpenCV libraries for computer vision. Judging by the many plaudits quickly garnered online when Vigasan revealed his build, he's nailed it, even if some people think it would be ideal for deadly accurate tennis ball volleys and other forms of sharp-shooting.

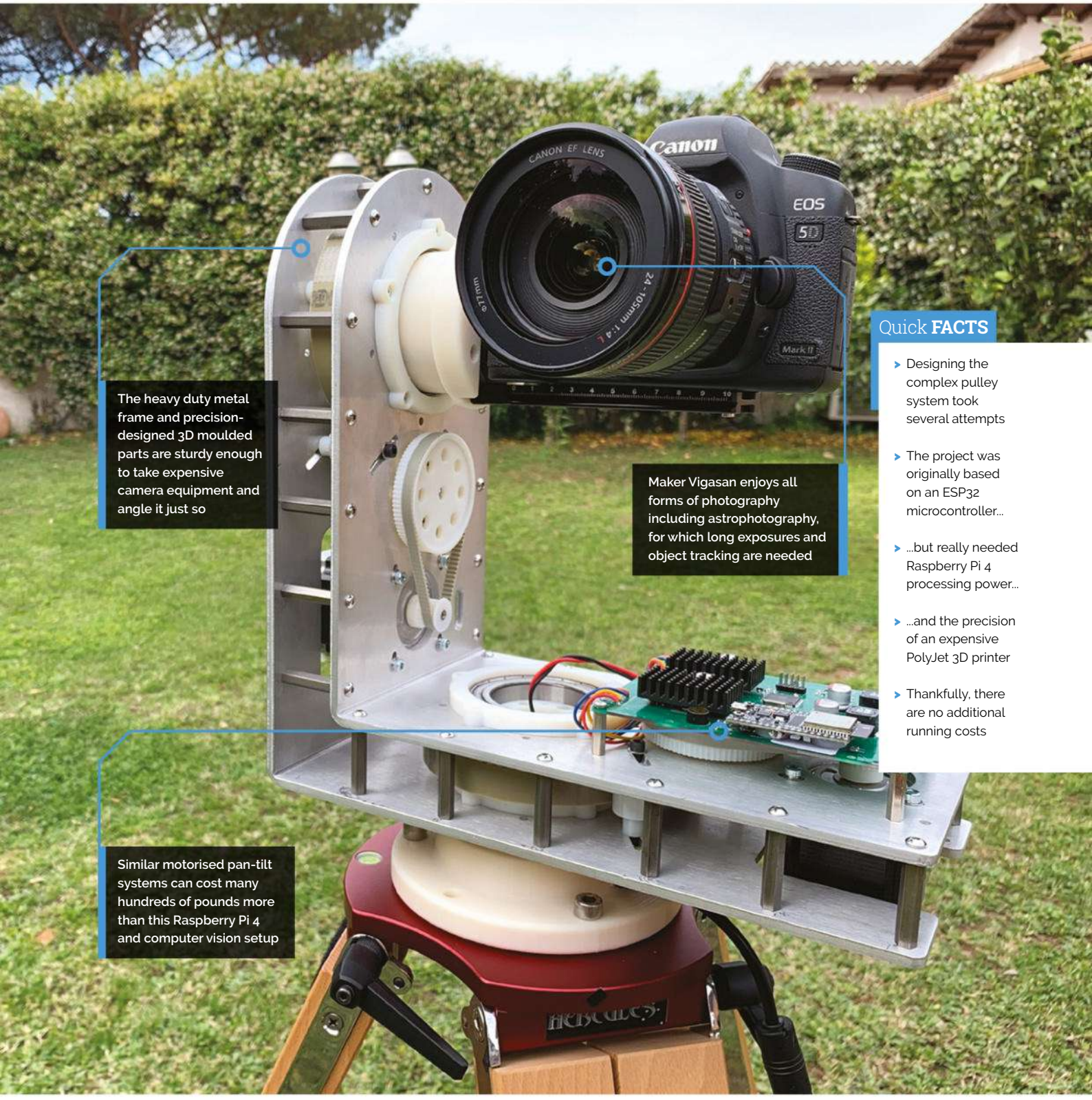
Labour of love

Not surprisingly, the Heavy Pan Tilt System – a "very robust motorised pan-tilt system capable of supporting cameras with heavy lenses" – is the work of many months. Its strength derives from the two aluminium frames, while Vito made everything else from parts he 3D-printed. He was lucky enough to have access to a fairly high-end PolyJet 3D printer, which meant he got very accurate results. Vito also designed the electronics and the control software, plus a custom circuit board, pairing a Raspberry Pi 4 and a small camera capable of automatic object tracking, as his video shows: **magpi.cc/pantiltyt**.

Vito began by creating a project using Raspberry Pi for object tracking, before moving on to the thornier issue of the mechanical design: "The need to move a camera with lenses of a certain weight requires considerable torque from the motors. So, to avoid using too big stepper motors, I designed a reduction system based on belts and pulleys with the advantage of using smaller motors."

Cost-benefit analysis

Although he can't quantify the time taken for designing the software and incorporating the necessary libraries (he's been an engineer for 20 years, so programming is part of his day job), Vito says the specific app to control things took him about a month, while designing the hardware took about twice that. Having initially tried to create the project using an ESP32 microcontroller and


CANON EF LENS
Canon
EOS
5D
24-105mm 1:4 L
Mark II
The heavy duty metal frame and precision-designed 3D moulded parts are sturdy enough to take expensive camera equipment and angle it just so
Maker Vigasan enjoys all forms of photography including astrophotography, for which long exposures and object tracking are needed
Similar motorised pan-tilt systems can cost many hundreds of pounds more than this Raspberry Pi 4 and computer vision setup

Quick **FACTS**

- Designing the complex pulley system took several attempts
- The project was originally based on an ESP32 microcontroller...
- ...but really needed Raspberry Pi 4 processing power...
- ...and the precision of an expensive PolyJet 3D printer
- Thankfully, there are no additional running costs

The mount can rotate a camera in any direction to track objects

The Heavy Pan Tilt System works incredibly smoothly

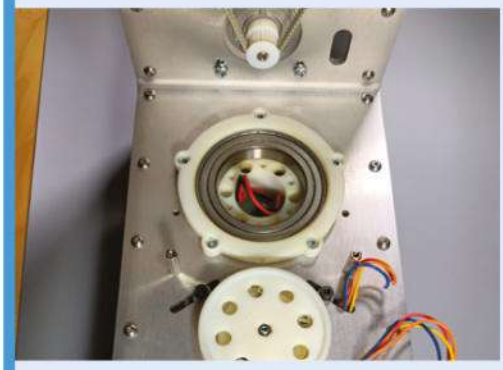
▲ One of the incredible photos Vigasan was able to capture using his pan-tilt DSLR camera setup

finding it lacked power, Vito was glad he switched to using Raspberry Pi 4, which "guarantees good computing power and the existence of a large number of libraries for any type of requirement."

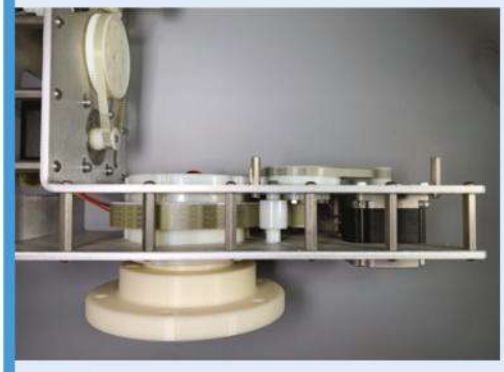
> "I don't think I created anything new, I just enjoyed doing it"

His choice of a rock-steady aluminium frame added €150 to the overall cost, which was in the region of €800. Even so, this is considerably cheaper than buying a specialist astrophotography rig. Vigasan modestly claims not to have invented anything: "I don't think I created anything new, I just enjoyed doing it and I hope that the project will be of help to those who want to use Raspberry Pi for their creations."

Pan, tilt, shoot!



01 Creating a motorised pan tilt system in order to take videos and photos using heavy cameras and lenses requires considerable torque, necessitating the use of stepper motors and a pulley and weights system to reduce the motor size required.



02 Vito designed and laser-cut the aluminium frames. These precision parts and other 3D-printed elements were designed to fit the two STM L6470 stepper motors and belts, which Raspberry Pi 4 controls.

03 Vito used Qt framework software and OpenCV libraries with Raspberry Pi OS for object detection and computer vision tasks. Everything can now be controlled via Bluetooth and an iPhone – see **magpi.cc/motorizedcamgit** for more details.

Word Clock

David Crookes reckons it's time we all paid some attention to Christopher Hall's amazing clock

MAKER

Christopher Hall

Christopher Hall is a senior network engineer from Pikeville, KY, USA. He's a maker at heart and lover of all things geeky.

magpi.cc/wordclock

If you want to know the time, there are many methods available to you. Perhaps you'd like to glance at a digital or analogue clock or watch. Maybe you'd like to ask your favourite virtual assistant to read it out loud. If you're feeling bold, you consult a sundial, a time ball or a merkhet. Or, if you're Christopher Hall, you could use a word clock and see the time as written sentences.

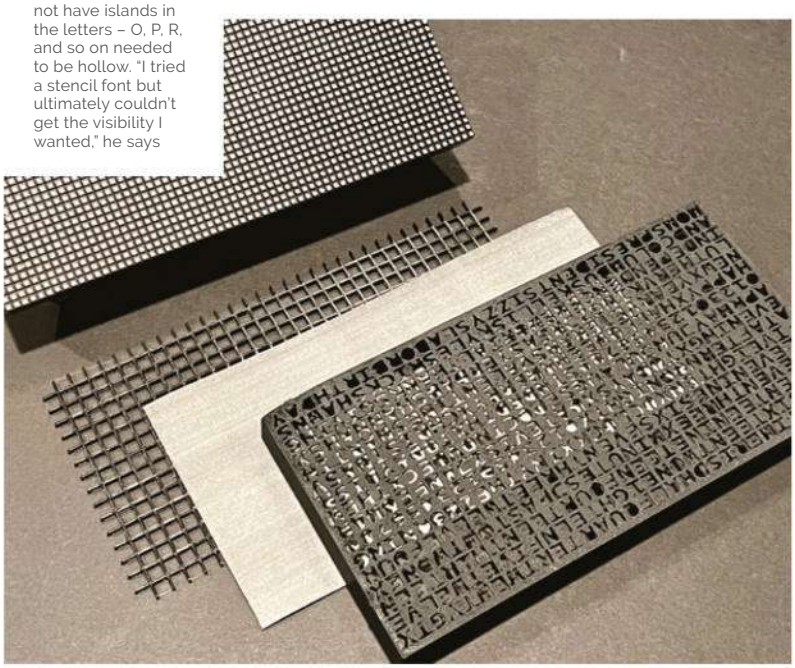
Word clocks are not new but they sure are fun, and Christopher's entry goes further than most. As well as telling the time, it announces holidays and birthdays by shining LED lights behind specific letters and numbers on a custom-made panel. Everything is controlled by a Raspberry Pi computer. "When I thought of the project, Raspberry Pi felt like a natural fit," he says.

Time to try

The idea came when Christopher bought a 3D printer. "Rather than printing figurines, I wanted something practical; something with function," he explains. "I was browsing Printables when I came across a similar project and thought, 'I bet I have most of the stuff I need to make this'." Even so, aspects of the build posed a challenge.

Since Christopher was making his own housing for the clock, he needed to use a 3D graphics tool. "That meant learning Blender and taking precise measurements with callipers," he says. He also opted to use a 128×64 HUB75 LED matrix that he'd received with a monthly HackerBox subscription, and this influenced the size of the word matrix that he had to print.

"There was a lot of trial and error," he says. "A key problem I noticed early on was light. I opted to use a 2×2 grid of LEDs per illuminated letter, but putting the letters too close meant I could see each light," Christopher says. "In the end, I designed a lattice to sit between the RGB matrix and the printed word matrix. Between it, I placed a grey sheet of paper cut to size, and this worked pretty well to diffuse the light."



Christopher discovered he could not have islands in the letters – O, P, R, and so on needed to be hollow. "I tried a stencil font but ultimately couldn't get the visibility I wanted," he says

Clocking on

Since the word matrix was 62×32 characters, Christopher realised he had space for many letters. "Once I'd finished arranging the time parts, I had more than half a screen's worth of real estate left, which is why I decided to also announce the date and any holidays and family birthdays," he says.

To keep time, Christopher included an RGB Matrix HAT + RTC from Adafruit so that the project could benefit from a persistent real-time clock. Setup involved connecting this to the Raspberry Pi computer, and hooking everything to the display inside the printed case.

Of course, Christopher also needed to program the device. "The project I found on Printables had an example Python script, so that was a good jumping off point, but it still needed to be heavily modified," he says. "I used a holidays library to determine if the current day was a holiday or not. I also had the challenge of lighting the letters."

For the sake of compactness, Christopher desoldered several components and reattached them directly to the board of the Adafruit RGB Matrix HAT

The HUB75 RGB Matrix is daisy-chainable so another screen could be attached, perhaps to display weather, email notifications, and network health issues

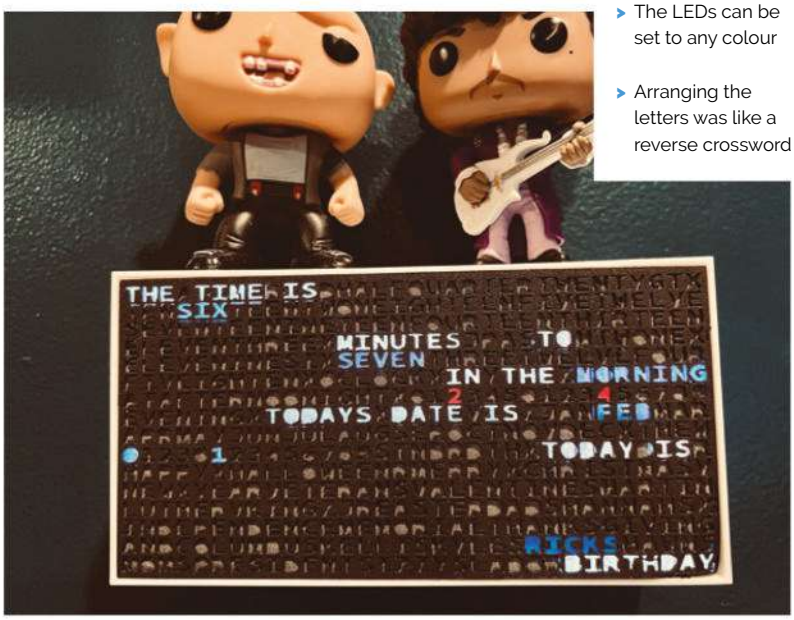
The case has been designed so that power can easily flow from a 5V power brick to the Adafruit HAT. From there, power goes to Raspberry Pi and the LED Matrix

Quick FACTS

- The clock highlights the letters needed for the time
- It displays holidays and birthdays too
- A shiny front was fixed with ultra-flat black primer
- The LEDs can be set to any colour
- Arranging the letters was like a reverse crossword

He had to figure out, for example, that LEDs at positions 14,46, 14,47, 15,45 and 15,47 needed to be turned on for the letter J. "This had to be done for each letter I wanted to turn on, so I set them into arrays for better organisation," he adds. Once done, the clock could spring into life, looping every second to update the LED clusters. "Raspberry Pi has been a great device for the job," he says.

> I used a holidays library to determine if the current day was a holiday or not



The completed Word Clock displaying the time, date, and a birthday. "Everything sits neatly in the 3D-printed, wall-mounted housing," Christopher says

Game Boy Interceptor

A challenge by fellow Game Boy Tetris fans led to this Pico-based recording device, discovers **Rosie Hattersley**

MAKER

Sebastian Staacks

Sebastian is a physicist and developer of the app 'phyphox' at the RWTH Aachen University (Germany). He is a father of two and presents his hobby projects in his blog...

there.oughta.be

The addictive qualities of the computer game Tetris are well-known, so it's perhaps no surprise to learn that there is a whole community of Game Boy Tetris fans who meet online for tournaments.

The RP2040 microcontroller-based Game Boy Interceptor came about when just such a tournament was being planned, "and, of course, they wanted to stream the contestants' gameplay," relates fellow Tetris fan Sebastian Staacks. "Streaming would not be a problem with a modified Game Boy or a modern Game Boy clone such as the Analogue Pocket," says Sebastian, "but it would mean contestants would be forced to use the same platform in order to compete." This change just wouldn't fly: "the contestants always played their favourite Game Boy model and, in a contest, would want to use the model on which they trained their muscle memory."

Getting everyone to modify their beloved handheld console was out of the question. Sensing a challenge he could relish, Sebastian agreed to work out a way of streaming the tournament that would satisfy everyone. His idea was to insert a device between the game cartridge and the Game Boy that checks what the handheld console is doing and reconstructs the gameplay from the data.

Game on!

Physicist Sebastian holds a PhD in solid-state physics, "which means that I know the basics of almost everything technical, but nothing that is required to apply it," he announces modestly. A self-taught programmer and electronics enthusiast since childhood, Sebastian follows his own advice that the best way to learn new skills is "to become obsessed with a project that is just a little bit above your current abilities."

When he got his first Raspberry Pi a decade ago, he wasn't quite sure what he would do with it. Sebastian has been running a Raspberry Pi-based home automation setup ever since, and also keeps a Raspberry Pi handy for the constant stream of projects that invariably need a simple server for IoT purposes. When Raspberry Pi Pico launched, he was curious about Raspberry Pi microcontrollers, especially after working with Arduinos for a while. Happily, "the RP2040 turned out to be a perfect match for GB Interceptor!" Unlike modern devices, the Game Boy reads data from the cartridge as fast as from its RAM, so there is no reason for it to load its code into RAM first. Instead, the code is executed directly from the cartridge (with few exceptions) and a device in between would know exactly what the Game Boy was doing.

Unexpected benefits

At first, Sebastian simply hoped to use RP2040 to capture information from Tetris, but the microcontroller was powerful enough to render the graphics, emulate the code from whichever game was being played, and act as a general-purpose video capture device. The power and sophistication of the RP2040 meant the Game Boy Interceptor was a much more useful and flexible device than Sebastian had anticipated. "You can simply plug it between the cartridge and your Game Boy and connect it to a host device via USB." There, it shows up like a webcam and, as a USB Video

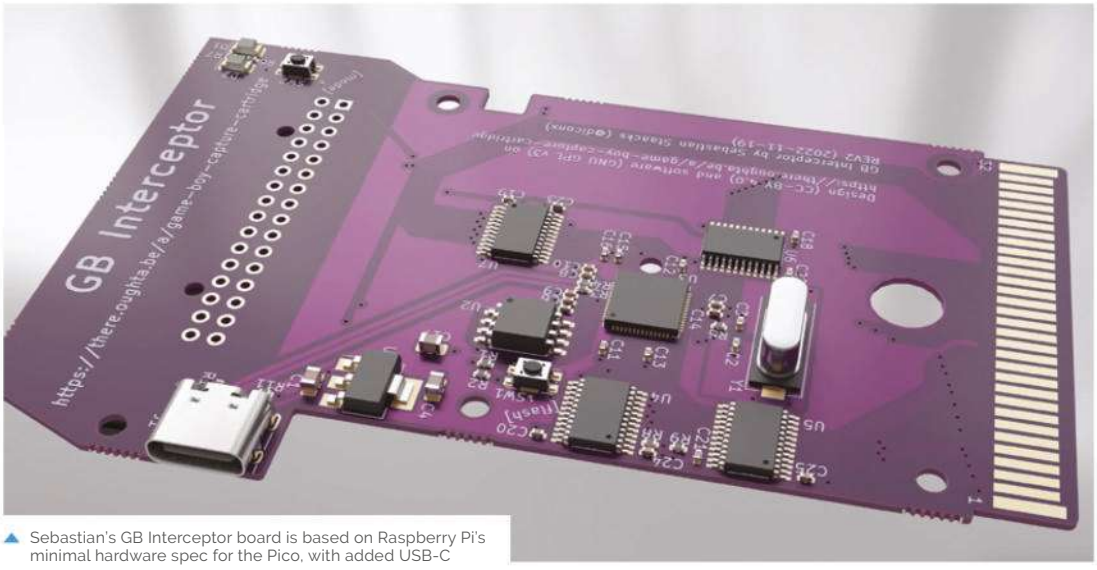
Sebastian realised that the entire game data would pass through the Game Boy's cartridge slot – and a powerful microcontroller could emulate it

The GB Interceptor acts as a video-capture device for older Game Boys, streaming moves during online Tetris tournaments

Raspberry Pi RP2040's GPIO pins and programmable IOs capture video RAM data and emulate moves without needing the gaming device to be modified

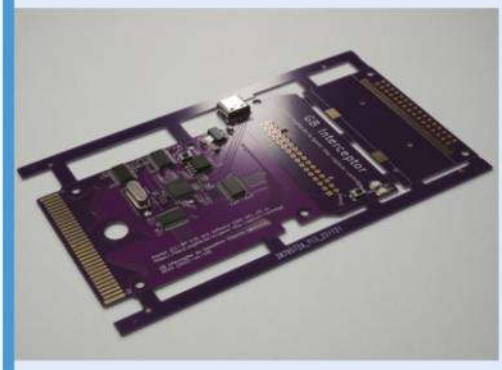
Quick FACTS

- The Game Boy has a 1MHz bus...
- ...and uses USB 1.1, with a frame rate of just 29 fps
- So, Sebastian had to be creative in replicating its effects
- Sebastian's LED Cube featured in *The MagPi* #100
- It has since been converted into an artificial fireplace!

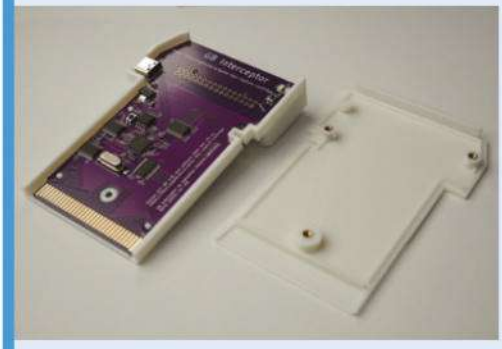


▲ Sebastian's GB Interceptor board is based on Raspberry Pi's minimal hardware spec for the Pico, with added USB-C

Capture the gaming action



01 The Interceptor works "like an emulator on rails" and executes the same instructions that the Game Boy receives from the cartridge to reproduce what the Game Boy writes into its video RAM.



02 Snug inside a 3D-printed case, the RP2040 emulates the Game Boy's graphics unit and renders images. "It works almost perfectly in 98% of the games tested," says Sebastian.

03 However, Sebastian had to overclock the RP2040 to 250MHz to get it to work. The Interceptor is currently limited to classical Game Boy games since Game Boy Color games use twice the clock speed.

▲ The GB Interceptor can even be used as a webcam, as Sebastian demonstrates

Class device, it does not need a driver and just works on Linux, Windows, and Android. It works with macOS too, though Sebastian says he still experiences some issues with M1- and M2-based Macs. It can even make use of the Game Boy's camera and function as a webcam.

> The design and implementation came together quickly

Sebastian had previously worked on a cartridge using the ESP8266 microcontroller, which had the benefit of Wi-Fi, but was far too underpowered to deal with the Game Boy's 1MHz bus speed (**magpi.cc/wifigbcart**). He had also undertaken a Raspberry Pi Pico project using its programmable IOs. Using RP2040 made sense since he didn't need wireless connectivity for the Game Boy Interceptor, whereas the PIOs were adept at reading bus data leaving the CPU free to pick up bus events at its own pace. This, plus the fact he uses Raspberry Pi "a lot", meant the design and implementation came together quickly. The sole changes from his first version were to correct the orientation of some of the LEDs and a switch to USB-C which is useful for indicating whether the Game Boy is on or off.

Escalate your controller design in style

Fully configurable universal case system- the perfect fit for all your application needs

Every panel on the UCS enclosure range can be customised enabling you to create your own individual stylish design easily.

With optional wall, desk & DIN rail mounting adaptors the range is available in 2 colours, 4 sizes & 2 heights ensuring you'll find the right enclosure for your next project.

For more information visit https://phoe.co/ucs-rpi-uk

PHOENIX CONTACT
INSPIRING INNOVATIONS

Aquarium dosing pump

There's something fishy about this fabulous project, but that's the whole point, as **David Crookes** explains

MAKER

Joe Stiff

Joe's life is powered by Raspberry Pi at this point and it's part of the ecosystem of his home. He promises more projects are coming soon!

mustardcorner.com

Keeping corals healthy is a tricky task. As well as requiring the right temperature and proper aquarium lighting, corals need a balanced healthy diet and extremely precise water chemistry. Joe Stiff, who keeps corals and macroalgae in 7.5-litre and 41-litre salt water tanks is fully aware of this. "Dosing various ions such as calcium and magnesium, as well as other chemicals, is necessary when taking care of corals," he says.

Rather than pay anywhere between £80 and £500 on a commercial dosing pump, however, he decided to make one himself. By using a Raspberry Pi Pico microcontroller, along with good-quality relays and pumps, he has been able to create a driver for a fraction of the cost – just £14 in total. He has also done so without compromising. "Price wasn't a limiting factor at all because these parts are relatively cheap," he explains.

Nothing fishy

Although he was always keen to keep costs under £30, the main requirement was to create a device that could dose a configurable amount of chemicals accurate to within 0.5 millilitres or less. He also needed each pump to separately operate a configurable number of times each day so that different liquids could be used at different quantities.

"The doser needed to be low voltage because of its proximity to water, and I wanted it to be small and neat so that it could go near a beautiful reef tank and not look out of place," Joe continues. "I also needed it to be repairable and easy to take apart if necessary to replace the parts. This is why I chose particular peristaltic pumps. You can take out the tubing and replace it with new tubing very easily from the front."

Peristaltic pumps are low-maintenance and easy to sterilise. They use rollers or shoes to effectively massage a fixed amount of chemical through the tube while preventing back-flow. Sourcing the pumps and the relay took Joe a few hours. He was then able to design a case using 3D modelling software, map the wiring and electronics on paper, solder a prototype, and write the software.

Good dose

"When you buy a commercial doser, you're stuck with the company's proprietary software which is usually awful and painful to use, or simply doesn't work half the time," Joe says. "This project fit my ultimate aim which is to build everything for the aquarium myself, from the LED fixtures to the doser and temperature/pH monitoring system."

Joe wrote the program in MicroPython and, after calibrating each of the doser's two pumps to run at the same speed, he set it to run every 15 minutes. This allowed a small amount of calcium hydroxide to flow 96 times a day from one pump,

> When you buy a commercial doser, you're stuck with the company's proprietary software

ensuring 200 ml of the chemical would be dosed on a daily basis. The other pump was set to dose other chemicals, such as amino acids and food, when needed.

It's certainly effective. Joe says that the Pico controls the relay, turning the pumps on and off with simple code sent via one of the GPIO pins. A few functions calculate the dosage/timing and the code just loops, sleeping in-between dosing. "My next step will be to upgrade to Raspberry Pi Pico W," he says. "I will then write some wireless LAN code to control it via my mobile phone."

The pumps are set to turn on for a specific amount of time: one second of dosing administers 1.43 ml of liquid. For small doses, the pump can be set for less than a second!

A Raspberry Pi Pico microcontroller sits within this beautiful 3D-printed case. The settings are currently being changed by connecting the doser to a laptop, but Joe wants to go wireless

Tubes can be easily attached to the front of the pumps which, in turn, are connected directly to a 5 V power supply. A signal from the Raspberry Pi Pico is needed to turn the switch on

Quick **FACTS**

- The device provides micro-doses of a chemical
- It's being used to keep coral healthy in a tank
- It could also be used to water plants automatically
- It could be adapted into a water change system
- Joe is considering selling ready-made devices

▲ The dosing heads cost just £5 each. Add Raspberry Pi Pico, a £3 relay, and £3 of wiring, tubing, soldering, plastic, and screws, and you have a very cheap dosing pump!

▲ Joe printed his case using a Voron 2.4 3D printer using glitter filament, allowing enough space for wiring and room for a charging port

Tank Driving Simulator

A Swiss tank museum is home to an historic training vehicle that has been upgraded with a Raspberry Pi. **Rosie Hattersley** finds out why

MAKER

Gerold Handschin, Michael Salathé, & René Demarmels

The museum team, including electronics specialist and technician Gerold, were determined to salvage the 50-year-old FASIP.

magpi.cc/fasip

Although it has become known by its German moniker of FASIP (shorthand for Panzerfahrsimulator), the vehicle was actually developed in France in the 1970s, explains Gerold Handschin, one of the trio from the tank museum who set about saving the unlikely survivor. "A total of eight tracks were used to train prospective tank drivers, initially for the Centurion tank and later for the Panzer 68 and Leopard 2," Gerold relates. The trainee driver sat in a replica driver's cab, with a screen displaying the terrain he needed to negotiate. Hydraulics under the cab replicated the tank's movements, while a camera on a trolley travelled the length of the 12m-long detailed terrain model. The combination of responsiveness to the driver's steering movements and on-screen display created a realistic driving experience that predated arcade games by several years. In addition to the large terrain model and hydraulic system, various computers were required to operate the system.

Ultimate upcyclers

In 2004, the Military Museum saved one of the FASIP systems in Thun from being scrapped, and rebuilt it at the museum site in the village of Full. But, after several years, various defects in the antiquated computers and in other electronic components had led to its apparent end. The Full museum's dedicated helpers, Gerold Handschin, Michael Salathé and René Demarmels, could not resign themselves to this. Instead, they began a thorough analysis of how the FASIP system worked, identified defects, and gradually replaced any components that failed. Their aim was to replace each item "as faithfully as possible."

Sadly, the original 1970s MITRA-125 computer which controlled almost every aspect, including responding to the displacement sensors and controlling the simulator's movements, lamp, and displays, was defective. This central control unit clearly needed to be replaced. "Fortunately, the program in operation on the MITRA-125 was available as a hard copy. So we decided to replace the old computer with a new system," says Gerold. The only issue: it was all in French. Scanning in the several-thousand-page document and using text recognition software helped, but in some cases the type had faded so much that the characters could not be recognised. "We sometimes had to approach the correct parameter values by trial and error," says Gerold, but new C-based simulator software for Raspberry Pi was eventually written.

Restoration challenges

The decision to use Raspberry Pi 3B+ as the new central control computer was down to its GPIO, its size, and the price. Raspberry Pi was fitted onto a new adapter board via its interface slot, replacing the interface board for the MITRA-125. The original power supply units were in poor condition and voltages were no longer stable, so modern switching power supplies were fitted. "The new adapter board contains multiplexers to connect all the signals needed for the Raspberry Pi GPIO," explains Gerold. It was also used to adjust the 3.3V voltage of the GPIO to the 5V TTL logic of the existing systems.

The FASIP tank simulator challenges the trainee driver to master eight different terrains, beamed to the cab from a camera trained on the scenic model layout

Originally running on custom software written for the MITRA-125 computer that powered it in the 1970s, FASIP now runs code written in C

This Raspberry Pi-controlled setup emulates the bumps and turns the driver encounters thanks to cameras and sensors that follow the tank simulator's route

Quick **FACTS**

- The team had four custom circuit boards made...
- ...just in case the tank simulator needed replacements in the future
- Overall, the project cost 820 Swiss francs, including spare parts
- The original instruction manual was in French
- So, they scanned in the text and used a translation tool

▲ View from inside the FASIP tank simulator control cab with live terrain updates as the driver navigates

▶ Sensors and camera hover above the test track, feeding back to the FASIP tank sim control deck

▲ The team looking over the scale model that the tank 'drives' around

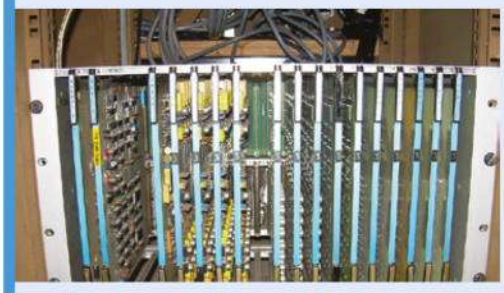
> A realistic driving experience that predated arcade games

The IO plug-in unit's original boards, XERUDI and XUCI, were replaced by the new board and Raspberry Pi too, but it was a challenge to work out the correct timing when driving the components of the interface plug-in unit. The overhaul of the simulator was completed at the end of July 2020. Since then, it has been possible to drive the Panzer 68 simulator through the grounds at the Swiss Military Museum in Full by appointment.

Tank training

01 The 1970s MITRA-125 custom code and hardware were replaced with a bespoke circuit board to control the cameras and respond to feedback from them and the track sensors.

02 The corroded old network and IOs needed to be replaced with components capable of delivering a constant voltage.



03 The original tank sim rig needed a complete overhaul, custom circuit board and adapters, plus reprogramming in C.

Progress Bar

Do you tend to keep an eye on the clock during the working day? **Nicola King** learns how watching for a rainbow is far more enjoyable

MAKER

Martin Spendiff & Vanessa Bradley

Martin is a mathematical modeller who left the UK for Switzerland, and a fan of FOSS and tech that serves users, rather than the people who made it. Vanessa is new to coding and a constant source of weird and good ideas.

veeb.ch

We have all, let's be honest, clock-watched at some point in our working lives. Well, friends of *The MagPi*, Martin Spendiff and Vanessa Bradley have come up with a little idea that might improve your working day, bringing a colourful rainbow display at close of business, where once there was the face of a benign and mundane clock. This physical Progress Bar (**magpi.cc/progressbar**) could be just the ticket to brighten up your office and gladden your soul.

Inspiration, Martin tells us, came from "a fondness for clocks and way too much time looking at progress bars. As a random bit of pub trivia… the progress bar was first used long before computers were invented – back in the 1800s."

So, Veeb's fresh take on the concept began percolating and, armed with a Raspberry Pi Pico W, a metre-long piece of frame from a hardware shop, a plastic light diffuser, and a one-metre, 144-LED, 5V addressable LED strip, the duo set to work.

A work in… progress

Martin and Vanessa quickly put the hardware together, wrote some MicroPython code, and then connected Pico W, which is the brain of the Progress Bar. "Pico W connects to the internet and finds out what time it is," reveals Martin. "After that, it checks the working hours (that you put into the code). It just does a little bit of arithmetic to figure out if you're at work, and how far through the day you are. It's then easy to figure out how many lights in the [LED] strip it should turn on."

Cleverly, the pair also incorporated a link to Google Calendar, so any appointments are shown on the bar (and constantly updated through the day), and the bar also flashes at appointment time as a vibrant reminder, although Martin envisages that maybe "a *Knight Rider*-style sweep would be better!" Linking to Google Calendar was admittedly "tricky", says Martin, but the gcalcli command-line tool (**magpi.cc/gcalcli**) handles all the Google authorisation.

The duo used the Wi-Fi-capable Pico W because "running a fully-fledged computer to turn on a few lights would be overkill," says Martin. "It needs to connect to the internet, so we [opted for] Pico W. It was the first time we've used one and it was surprisingly easy."

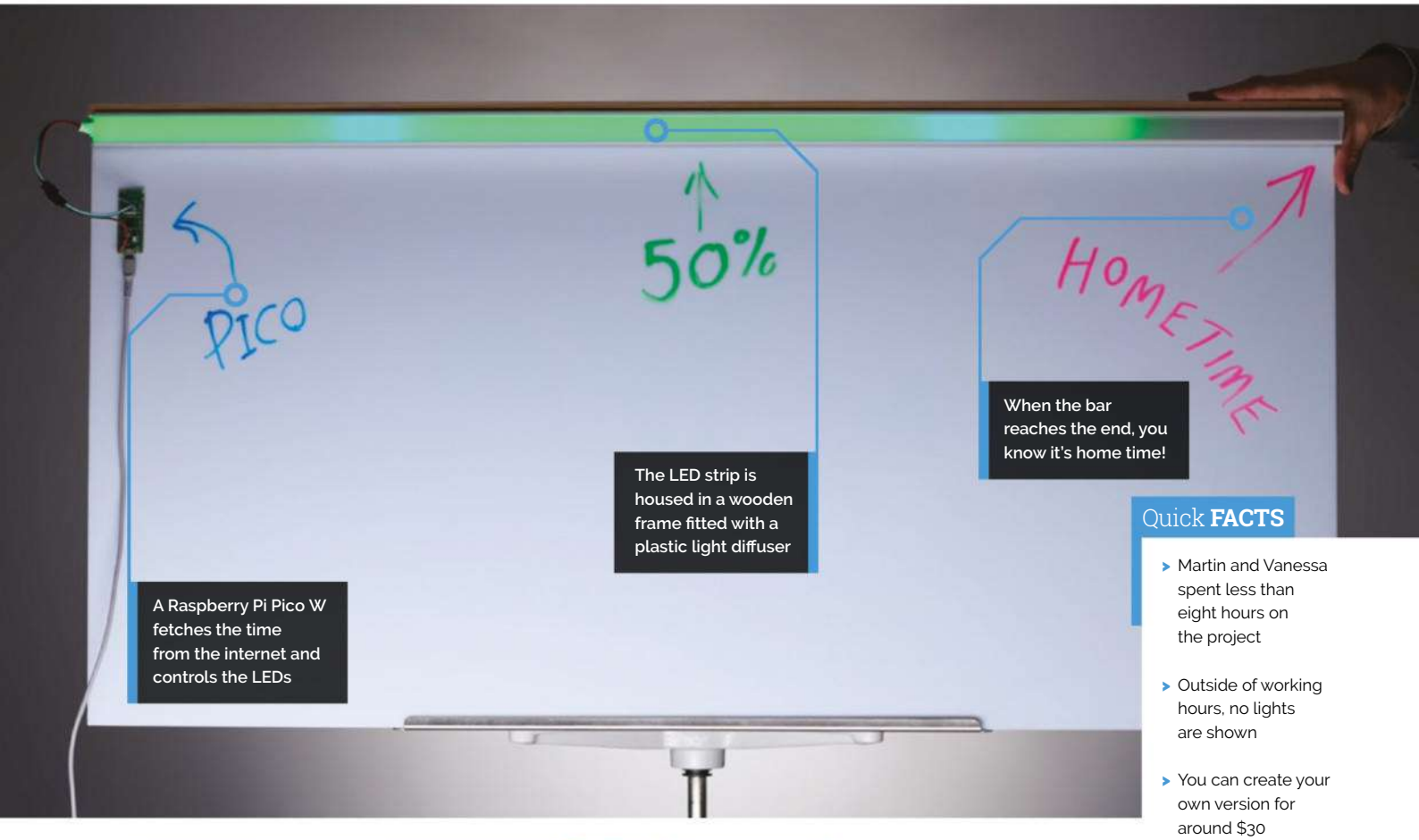
He and Vanessa are always open to suggestions for tweaks – when someone saw the GitHub page and wanted to use it in a school without the Google Calendar link, they were keen to help, "The idea of a classroom getting to see a rainbow at home time was sufficiently appealing to make us refine the code."

> "It needs to connect to the internet, so we opted for Pico W"

Easy as Pi

One of the great features of the Progress Bar is its relative simplicity, with very little needed in terms of hardware, making it an ideal project for any new makers. "It is a nice little example that involves very little soldering and relatively simple code," Martin highlights.

He and Vanessa have received highly positive feedback from the maker community – although Martin recalls how one person did declare, 'imagine hating work so much that you need to keep looking at one of these.' The pair are pragmatic: "If being pleased it's home time is a crime, guilty as charged!" They use their Progress Bar every day, highlighting, "There's a lovely element of it gently reminding you to work and when to down tools for the day."

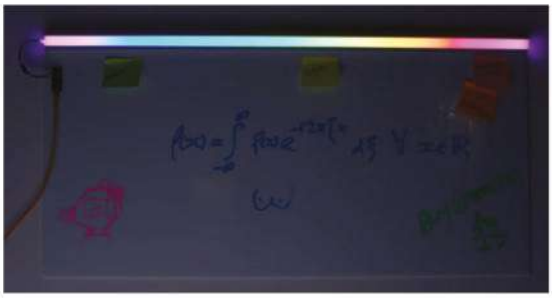

PICO
50%
HOMETIME
When the bar reaches the end, you know it's home time!
The LED strip is housed in a wooden frame fitted with a plastic light diffuser
A Raspberry Pi Pico W fetches the time from the internet and controls the LEDs

Quick **FACTS**

- Martin and Vanessa spent less than eight hours on the project
- Outside of working hours, no lights are shown
- You can create your own version for around $30
- The code is on GitHub: **magpi.cc/hometime**
- The code can be edited to reflect your own start/end time for each day

◀ Adding a plastic diffuser strip softens and blends the light from the LEDs

▼ Fitting the Progress Bar to the top of a whiteboard



▲ The colours of the rainbow signify that work time is over

BrachioGraph

A charmingly inaccurate plotter driven by Raspberry Pi draws the interest of **Rosie Hattersley**

MAKER

Daniele Procida

Daniele is head of engineering at Canonical, and a regular presenter at Python conferences around the world.

magpi.cc/brachiograph

Many of us remember being shown, or even making, pantograph pen plotters from the days before graphing tools were part of a standard home or office computing feature set. Open-source computing guru Daniele Procida recalled just such a device when considering a low-tech example of what Raspberry Pi could be used for while planning for the PyCon Namibia conference. The result, BrachioGraph, which sketches portraits and reproduces images of objects, was an instant hit, inspiring a fan following of keen makers.

Noting the contrast between the resources typically available at conference events in the UK and Europe and those in African host nations, Daniele says: "It's one thing doing things with robotics if you've got access to 3D printers, laser cutters, and off-the-shelf parts and components; quite another if you're an undergraduate at the University of Namibia, or a high school student and probably have to share a laptop or computer." Combined with Daniele's own interest, but lack of experience in robotics, he decided "to do something that was as affordable as possible, but also meaningful and also accessible, that didn't require an absolute minimum of materials and knowledge and facilities."

Motion capture

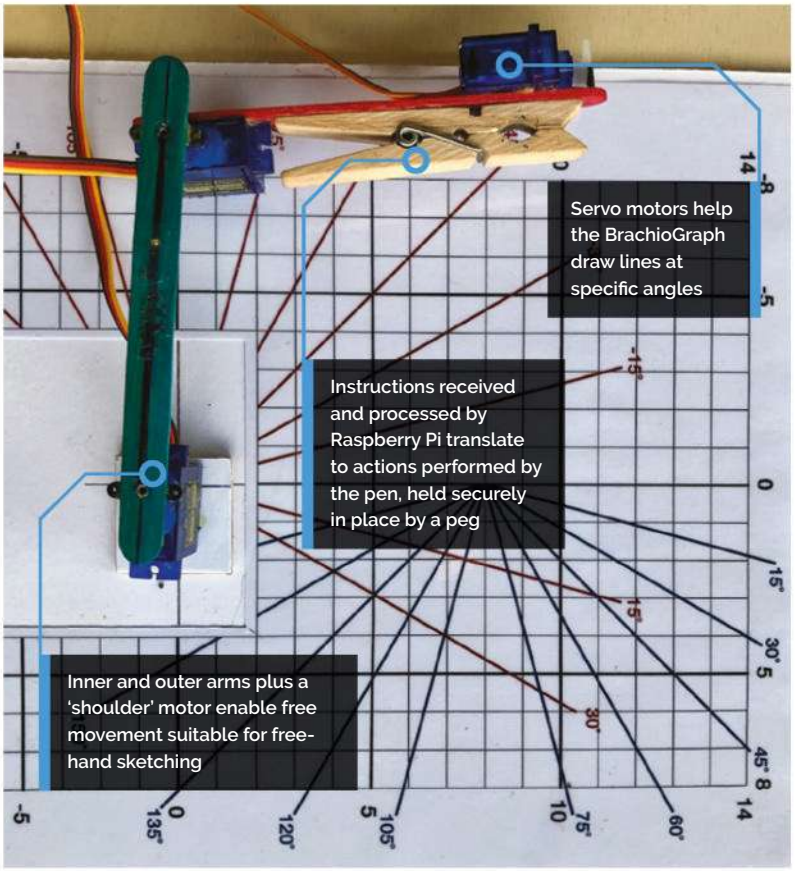
Playing around with servos and Raspberry Pi was the entry point for Daniele's plotting device project. Moving them around was easy enough, but he was keen to see a practical result: "What can we actually do?" With nostalgia for pantographs and XY aerial plotters in common, Daniele explains he found "their ability to reproduce something we do with our hands intrinsically fascinating. When you see a pen moving around, it looks like it has a purpose".

Although servo motors can be used to drive a device along straight lines, translating the movements and hand-drawn lines on a page is more about angles. The question, Daniele says, was whether he could turn movements into lines on a page. "Of course you can, because your own arm, if you're drawing with a pen, also doesn't have motors that go in straight lines. It just uses angles." He began by modelling the design on that of a pantograph, which faithfully mimics the movement of the human arm. A BrachioGraph, meanwhile is an 'arm plotter'.

Copycats encouraged

The BrachioGraph can be set up to run on its own, making it an ideal conference exhibit. "At one conference, I was very embarrassed because I

From bitmap to plot via vectorisation, the Prague Conservatory concert hall


Servo motors help the BrachioGraph draw lines at specific angles
Instructions received and processed by Raspberry Pi translate to actions performed by the pen, held securely in place by a peg
Inner and outer arms plus a 'shoulder' motor enable free movement suitable for free-hand sketching

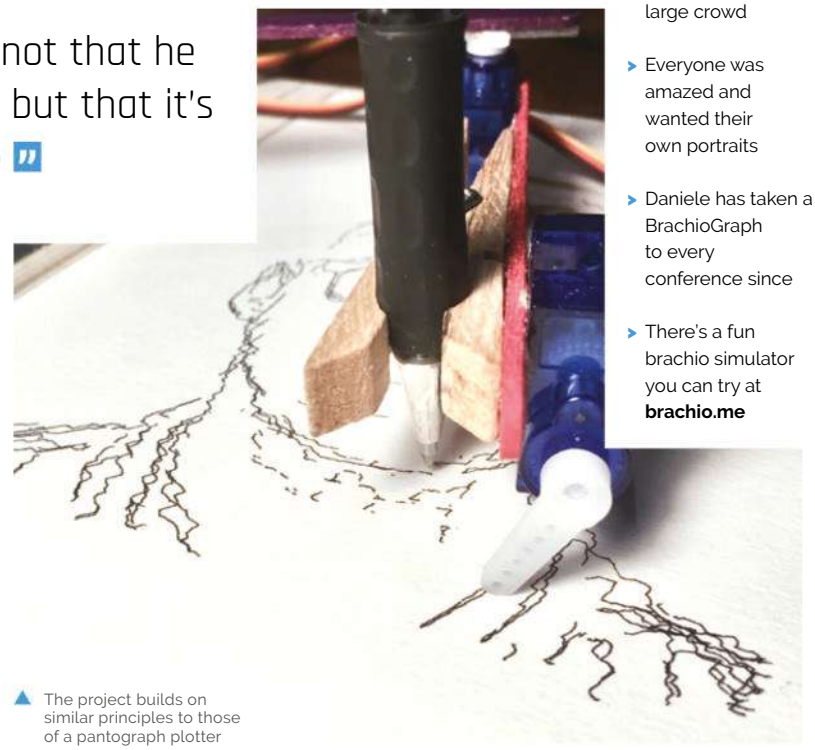
▲ Given an image to copy, the BrachioGraph is able to reproduce it

> The most important thing is not that he built this thing, and it works, but that it's reproducible by other people

borrowed a table in the sponsors' hall and just left something running, plotting." When he returned, the crowd was three or four deep around the table where the BrachioGraph was in action, taking away all the attention from sponsors who bought tables.

Daniele says that, after lots of design drafts and experimentation, the most important thing is not that he "built this thing, and it works," but that it's reproducible by other people and with household materials such as ice cream sticks, or cardboard and pencil, or clothes.

"The nicest thing for me is that, every so often, somebody will email me and say, 'Oh, it was my 12-year-old niece's birthday and, when I was visiting, I took a Raspberry Pi and three servo motors, and then we spent the weekend 'making a BrachioGraph'."



▲ The project builds on similar principles to those of a pantograph plotter

Quick FACTS

- Daniele unveiled his BrachioGraph at PyCon Namibia in 2019
- It was "a big hit", attracting a large crowd
- Everyone was amazed and wanted their own portraits
- Daniele has taken a BrachioGraph to every conference since
- There's a fun brachio simulator you can try at **brachio.me**

Herbie_Bot

Keeping weeds at bay is no easy feat – unless you have a robot to help you. **Rob Zwetsloot** slaps on some suntan lotion and has a look

MAKER

Russ Hall

An electronic technician for the US Postal Service, servicing high-speed mail processing machines.

magpi.cc/herbiebot

"Have you ever lugged around the yard a gallon jug of lawn-safe herbicide with a battery-powered spray wand?" asks Russ Hall. He's the creator of many lawn-keeping robots, and now is battling against pesky weeds.

"After a while, the hands, arms, and eyes tire of the repetition; it's work!" Russ continues. "I remembered my yardbot/mowbot, and thought I could rig up a sprayer with a camera and AI."

Russ had previously built a mapping robot for inside his house using Raspberry Pi, and decided to use that instead of the ZED camera and Nvidia-powered computer. "Raspberry Pi was easier to work with, as well as being less expensive," he says.

Yard work

"Working with hardware is a fun challenge to me and this robot was no different," Russ explains.

▲ It's much easier to get a robot to lug around a big bottle of herbicide

"These small battery-operated spray wands are versatile, and I thought it would be easy to fit into the robot. Unfortunately, my welder was stolen from me, so I used wood for the bracing and construction, built on the steel rover chassis that I had from years ago."

We actually quite like the look of the finished product – it's like a classic Rube Goldberg machine you'd see in a cartoon, in its own way. Not many Rube Goldberg machines employ modern machine learning though.

"This was my first AI project, so I had a lot to learn," Russ continues. "However, with the OAK camera and its Movidius chip, I felt it would work well with Raspberry Pi, since the AI is handled in the camera. To start, I used a standard routine in the Raspberry Pi to take photos of the lawn and weeds by the robot, because it's important to train the AI with photos that are identical to the video stream that it will be expected to process. I used Roboflow to prepare the images, and YOLOv5 to do the AI training (on Google servers), and it wasn't too hard. Luxonis has a neat conversion tool online, to make the software blob that is loaded in the camera."

Weeds: terminated

The video of Herbie in action can be viewed at **magpi.cc/herbiebot**, and it looks great at work – precisely targeting weeds and giving them a quick spray.

"If the weeds are too heavy, it can't hit them all, of course, since the robot is constantly moving forward," Russ says. "Testing with blue dye in the herbicide confirmed the accuracy and proper operation. I used 6 VDC for the sprayer servo so that it is quick. The 5 VDC for the sprayer pump comes from the USB power." Currently, Herbie is not fully automated to cover the yard, but it's in the works. Maybe one day the all-in-one lawn-keeping robot will be reality.

A camera scans the lawn to spot weeds using a model trained on the grass it's scanning
Parts of the rover are powered by a car battery, with several other battery sources employed
This battery-powered herbicide spraying wand is controlled via Raspberry Pi

Quick **FACTS**

- Russ builds robots with his kids
- Herbie is powered by Robot Operating System (ROS)
- The spray wand has 15 set positions...
- ... and is only 12 inches (~30 cms) behind the camera
- It runs on a Raspberry Pi 4

> It's important to train the AI with photos that are identical to the video stream that it will be expected to process

▲ Blue dye was used to test the accuracy of the model

▲ The spray wand is attached to a servo that Raspberry Pi moves to detected weeds

Warning!
Power tools and herbicides

This project involves woodworking which can be very dangerous, and herbicides that can be poisonous to humans. Exercise caution if you plan to work with both.

magpi.cc/diysafety

CinePI

Using Raspberry Pi as a high-end film camera is not as weird as it may sound. **Rob Zwetsloot** gets the shot

MAKER

Csaba Nagy

A student at the University of Alberta, who is also a maker exploring the intersections of film and technology among his other passions.

magpi.cc/csabanagy

Whenever a new Camera Module, or updated software for existing camera hardware comes out, we dream of all the cool things we could do with it. While custom cinematography and automated processes are high on our list, making a cinematic-quality camera is where Csaba Nagy went with CinePI.

"CinePI is an open-source project that can transform a Raspberry Pi into a high-end cinema camera," Csaba tells us. "Its standout feature is the ability to record 2K RAW Cinema DNG video at frame rates up to 50 fps with 12-bit colour depth. This makes CinePI an ideal camera for capturing cinematic-quality footage in various projects, such as short films, commercial work, and YouTube videos."

2K is basically 1080p. However, at a 12-bit colour depth, that is astonishing – most stuff you see will be at 8- or 10-bit colour.

"Almost a decade ago, I began using a professional video camera, the Blackmagic Cinema Camera, for my film-making hobby," Csaba continues. "The steep learning curve that accompanied the camera forced me to gain an in-depth understanding of the processes involved in creating cinematic images, aspects that most consumer and amateur camera gear obscure from users in the name of convenience and simplicity. As a result, I spent countless hours learning about image sensors, codecs, bit-depth, dynamic-range, colour theory, and other related concepts, which allowed me to gain a deeper understanding of camera systems and how they function."

> "It's competitive with cameras that cost orders of magnitude more than the parts required to build a CinePI"

Using this knowledge, Csaba went about trying to build a low-cost, open-source camera that everyone could use.

Software first

Csaba chose Raspberry Pi for this project due to community and software support that comes with it – apparently some of the code is based on stuff that started life on official projects and tutorials.

"The development/build process for the most recent version of the CinePI began in September of 2022, where initially the core software was being tested/developed," Csaba says. "The physical/hardware design quickly came together in January of this year, where the body was 3D-printed and all bits of hardware were assembled together into the final end-result.

CinePI records incredibly high-quality RAW footage that is great for editing
Using a Raspberry Pi HQ Camera and a good lens allows for great footage
This 3D-printed case holds everything in and is designed to be used like a camera

Quick FACTS

- Future exposure assist tools would include histograms, zebras, waveforms, and more
- For the cinematography buffs, CinePI fills a gap between Apertus Axiom Beta…
- …and Canon cameras with Magic Lantern firmware
- The 12-bit colour depth allows for extensive colour grading in post-production
- It makes use of a Raspberry Pi HQ Camera

"There were many modifications that were made to individual components in order to have all the hardware co-operate with each other. The greatest hurdle to overcome was the use of a HyperPixel 4.0 Square display which utilises nearly every GPIO pin of Raspberry Pi, meaning all extra hardware would need to interface with Raspberry Pi using a single I2C bus. This included an RTC, an I2C I/O expander, and power management board, which all had to connect via this single bus."

Quiet on set

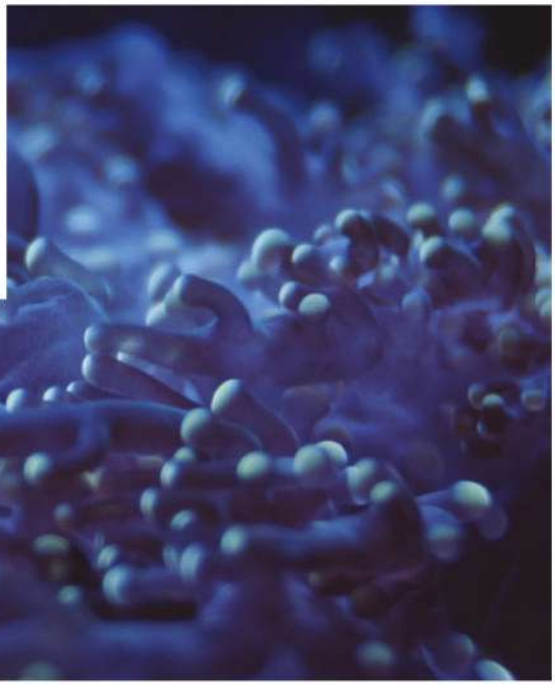
The result not only looks very cool but is incredibly functional, being able to record RAW video at 50 fps and 12-bit colour depth – remember that films run at 24 fps, so this makes for an excellent working file.

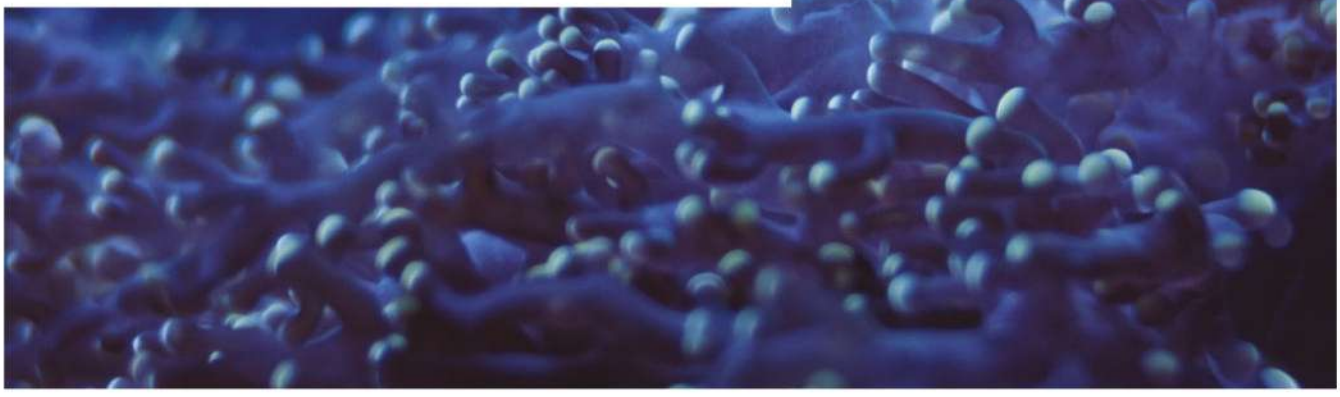
▲ You have full access to Raspberry Pi ports, as well as other specialist recording ports

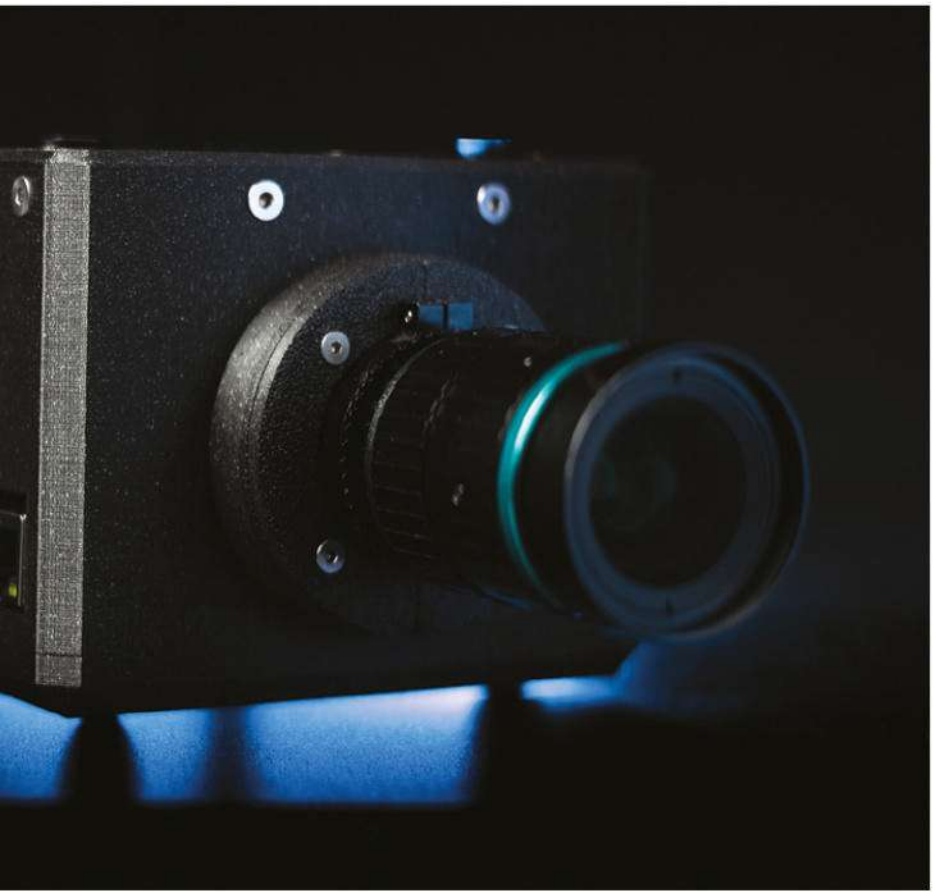
▲ Footage of an aquarium has been used in tests

◀ All the advanced options are surfaced for you to tweak with

▼ Shadow and colour data is available in RAWs, so you can change the look of your footage easily







▲ Not only is it very functional, it also looks great in our opinion

"It delivers better image quality than the Raspberry Pi's built-in image processing engine, which uses H.264 compression," Csaba explains. "Users can customise the camera with their own external sensors, buttons, HATs, and I/O. Real-time image monitoring and adjustments are possible with an attached HDMI monitor or compatible LCD touchscreen… CinePI records video as hundreds of frames in Adobe Cinema DNG format, which can be edited using software like DaVinci Resolve.

"The test footage I have captured demonstrates that the 12-bit RAW codec retains a significant amount of colour information in the shadows and highlights, resulting in superior quality compared to what the Raspberry Pi can offer by default… It's competitive with cameras that cost orders of magnitude more than the parts required to build a CinePI."

Csaba is not done yet, and wants to add a larger sensor, a HAT that handles a lot of the extra functions, and exposure assist tools – stuff that is beyond our understanding in general, and beyond Csaba's skills to currently implement.

"[I'm] always looking for more people to get involved with the project," he finishes – head to **cinepi.io** if you want to help.

Art house filming

01 It's not quite as easy as point-and-shoot, but that's by design! Using the touchscreen, you can confirm your settings before starting to record with the physical button up top.

02 Film is recorded RAW at extremely high quality, allowing you to colour-grade normally, or even set your own specific colour tone for your project.

03 You'll need to play back and edit on a PC, but that's easy enough to do if you use a USB 3.0 stick to record onto in the first place.

SUCCESS STORY magpi.cc/success

Yoto Player

The development possibilities of Raspberry Pi enabled an appealing, child-focused audio player to take shape

Yoto Player provides a great experience that children own and control, without introducing more screen time. Thanks to Raspberry Pi and the support and expertise of an approved reseller, it's a UK success story.

Ben Drury and his Yoto co-founder Filip Denker both have backgrounds in digital music services. Just as they were looking to set up a new venture, both were becoming parents for the first time. This led them to consider how young children interact with technology, and they decided to focus on a product that allowed kids to access technology without using a screen or needing a parent to operate it for them.

THE CHALLENGE

"Audio is better than screen time for inspiring creativity and imagination," Drury says of the company's approach. With children at Montessori nurseries, the idea of a physical interaction-based learning environment came about. But digital entertainment for children seemed to be predominantly based around phones and tablets, with young children increasingly spending time on platforms such as YouTube and on dedicated apps for kids. Drury and Denker's key idea was to develop an audio entertainment product that children could use without the need for yet more screen time.

THE SOLUTION

"Yoto Player is designed to be a physical way that kids can be in control of their own listening experience using physical NFC cards," explains Drury. "Kids enjoy collecting and ordering cards – so it seemed an obvious use of them for storytelling and podcasts."

Previously, Drury had used Raspberry Pi for monitors and servers, but not for standalone connected devices. He began using the Pimoroni

8×8-pixel Unicorn HAT (later, a 16×16-pixel one) and, based around this, hashed out a product form in a makerspace workshop. They also "wrote loads of code to run on Raspberry Pi and server-side, and also the Yoto app for iOS and Android."

R&D and PCB engineering and design for the first product, as well as assembly, were all done in the UK. The first product iteration was based around Raspberry Pi 2, switching over to Pi Zero and eventually Raspberry Pi Zero W when it launched – a "huge thing" for Yoto, as it meant they could build a wireless version of their storytelling player. "We were concerned by the availability of the Pi Zero W boards, as they were quite new at the time, but we took the risk and we were able to secure supply," says Ben.

> Without Raspberry Pi, we wouldn't have been able to build it in the first place

WHY RASPBERRY PI?

Drury is a long-time enthusiast of British computing and engineering, with fond memories of his BBC Micro, Acorn, and Archimedes computers. He was intrigued from the outset about Raspberry Pi, and was an early investor in Pimoroni, a long-standing Raspberry Pi Approved Reseller which later became Yoto's development partner. This was an inspired move, adding hardware know-how to Denker and Drury's expertise in software and user interfaces.

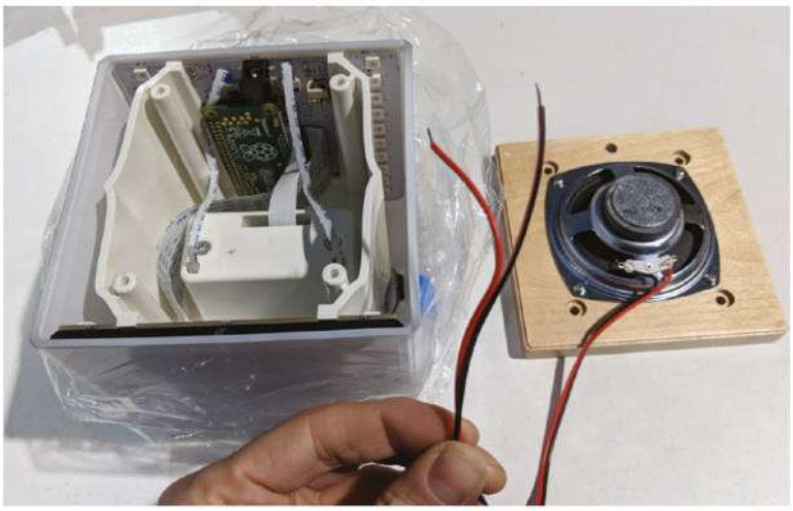
December 2017 saw Yoto's first Kickstarter launch, with shipping of their initial run of 750 units in late 2018. A few issues with EMF leakage meant some design tweaks were needed and the Yoto team were able to benefit from their strong, well-established relationship with Pimoroni, calling again on their technical expertise.

THE RESULTS

The first Yoto Player met with success, launching with a starter pack of stories, including ones by Roald Dahl, as well as blank cards for kids to record their own stories onto the player's 8GB SD card. Capturing the imagination of Roald Dahl's grandson early on – he's now a Yoto board member – cemented interest in the device's concept, and a whole slew of children's publishers followed. Julia Donaldson's beloved *The Gruffalo*, as well as some of Enid Blyton's extensive bibliography, are among the most famous.

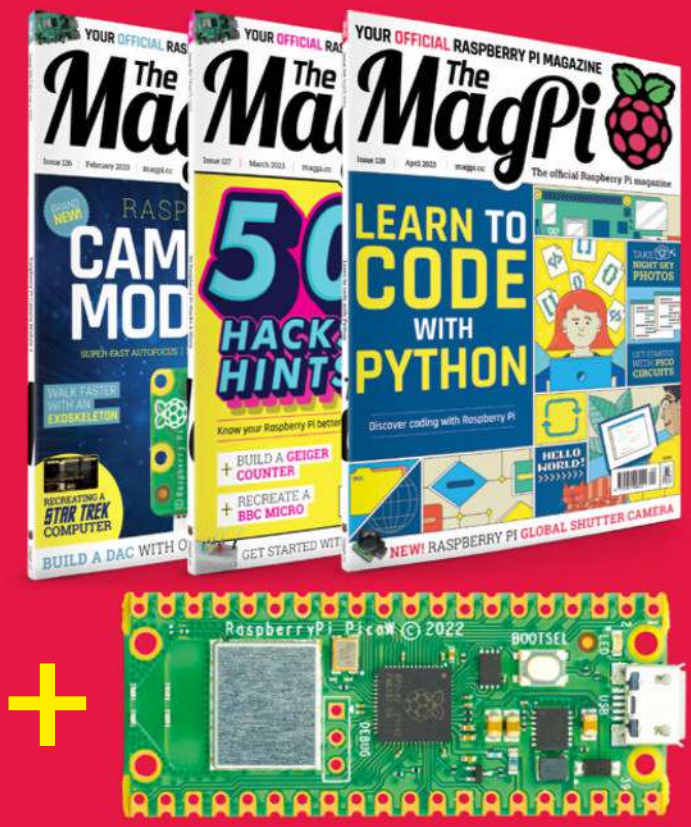
There are also phonics cards and 100-word visual cue cards for learning vocabulary from five other languages. With multilingual children, Ben and Filip are strong advocates for early language learning. They see Yoto Player as the sort of device that can help children who don't have such ready access to a wide vocabulary at home to begin acquiring a broader verbal language base.

Raspberry Pi was "absolutely crucial" to the development of Yoto Player, says Drury. "Without Raspberry Pi, we wouldn't have been able to build it in the first place."



SUBSCRIBE TODAY FOR JUST £10

Get 3 issues + FREE Pico W



Subscriber Benefits

- **FREE Delivery**
  Get it fast and for FREE
- **Exclusive Offers**
  Great gifts, offers, and discounts
- **Great Savings**
  Save up to 35% compared to stores

Subscribe for £10

- Free Pico W
- 3 issues of The MagPi
- Free delivery to your door
- £10 (UK only)

Subscribe for 6 Months

- Free Pico W
- 6 issues of The MagPi
- Free delivery to your door

£30 (UK) £35 (USA)
£35 (EU) £45 (Rest of World)

Subscribe by phone: **01293 312193**

Subscribe online: **magpi.cc/subscribe**

Email: **magpi@subscriptionhelpline.co.uk**

Subscribe for £10 is a UK-only offer. The subscription will renew at £15 every three months unless cancelled. A free Pico W is included with a 6-month subscription in USA, Europe and Rest of World.

SUBSCRIBE TODAY AND GET A

FREE Raspberry Pi Pico W

Subscribe in print today and get a **FREE development board**

WORTH $6

- A brand new RP2040-based Raspberry Pi Pico W development board
- Learn to code with electronics and build your own projects
- Make your own home automation projects, handheld consoles, tiny robots, and much, much more

This is a limited offer. Not included with renewals. Offer subject to change or withdrawal at any time.

YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
The official Raspberry Pi magazine
LEARN TO CODE WITH PYTHON
Discover coding with Raspberry Pi
HELLO WORLD!
NEW! RASPBERRY PI GLOBAL SHUTTER CAMERA

Buy now: magpi.cc/subscribe

SUBSCRIBE on app stores

From **£2.29**

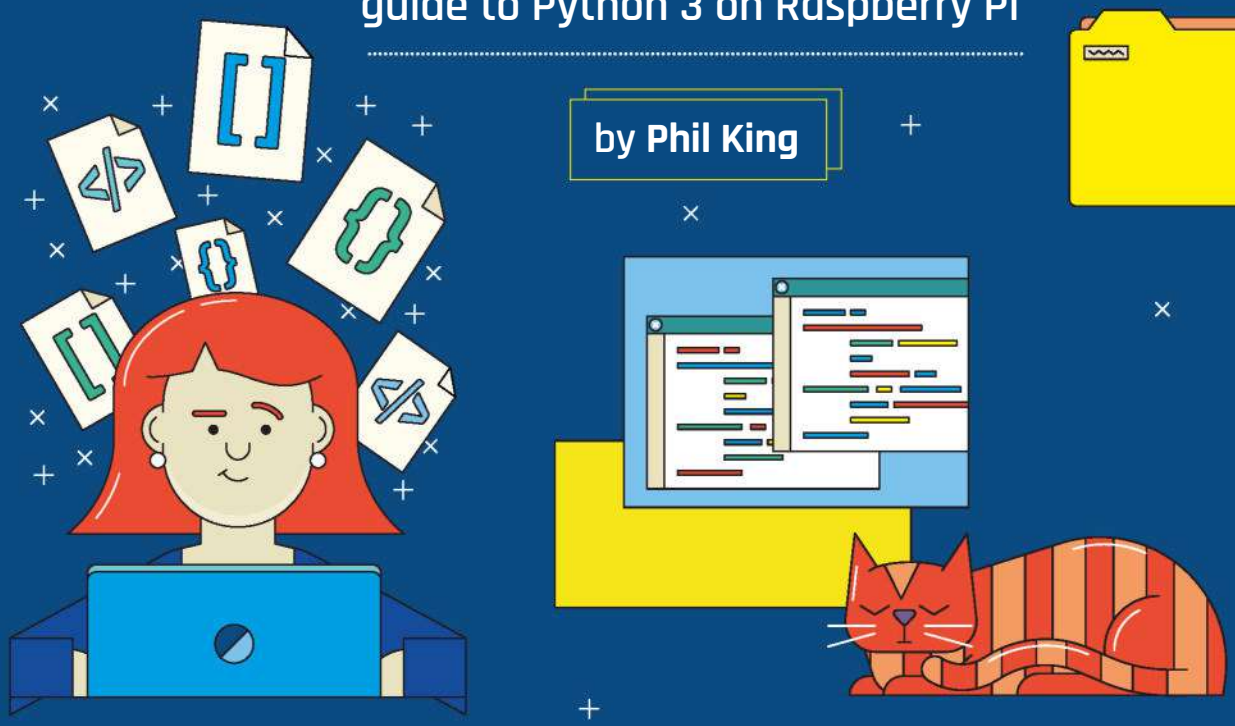
Available on the App Store

GET IT ON Google Play

LEARN TO CODE WITH PYTHON

Discover the joy of coding with our beginner's guide to Python 3 on Raspberry Pi

by Phil King



In this crash course, we'll take you through the key principles of Python programming on Raspberry Pi.

For this guide, we'll be doing all our coding in the Thonny IDE (integrated development environment) in Raspberry Pi OS. An IDE is a program that is used to create other programs. You can create programs in any text editor, but using an IDE makes life easier (especially for newcomers).

From the desktop applications menu, select Programming > Thonny. The Thonny IDE starts up in simple mode by default. Along the top are large icons including Load, Save, Run, and Stop.

Let's write some code

The main panel is where you will write your lines of Python code. At the bottom is the Shell panel where you'll see output and any error messages resulting from a running program.

Let's write some code!

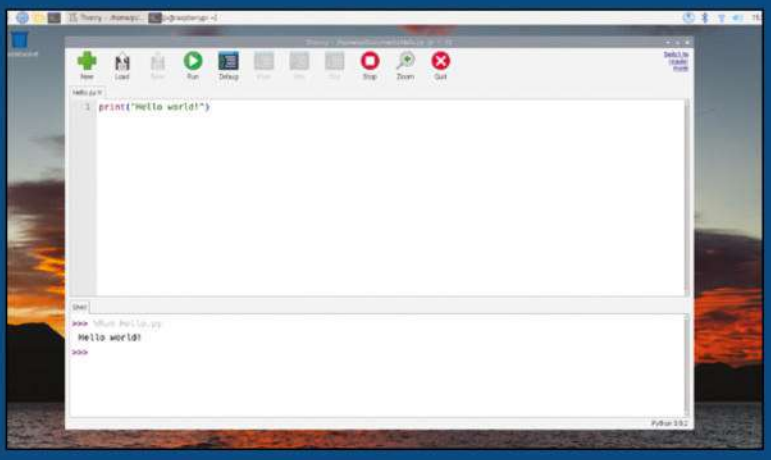
For our first program, we'll print a message to the Shell. This is the window at the bottom of Thonny that displays the results (or output) of our program. It's tradition to welcome a new programming language by getting it to say "Hello, World!"

In the main Thonny panel, on line 1, add the following Python code:

```
print("Hello, World!")
```

The double quotes are used to enclose a text message (known as a string). Click the Run icon and Thonny will run and print the message to the Shell. Try changing the text between the double quotes and rerunning it to see a different message.

▲ Numbers and text strings can be stored in variables and then retrieved when needed



▲ Hello world: the simplest of programs is to print a message to the Shell area

You can save the program by clicking Save and entering **hello_world.py** in the Name field.

Date types and maths operators

In our **hello_world.py** program, we printed a string of text. Other data types in Python include integers (whole numbers, such as 1, 2, and 3) and floating-point numbers (decimals with a point, such as 3.5 or 10.532).

To create these, we omit the double quotes in our print statement. Click New to start a new program and enter:

WHY PYTHON?

There are lots of different programming languages around, but Python is by far the most popular for beginners. Here are five reasons why you should probably choose Python as your first language:

1. **It's simple:** Python has a relatively simple and easy-to-learn syntax.
2. **Python is versatile:** Python can be used for a wide range of applications, from web development to data analysis and artificial intelligence.
3. **Large and active community:** Python has a large and active community of developers who contribute to its development. This community ensures that there are many resources available for learning and troubleshooting.
4. **Open-source:** Python is an open-source language, which means that the source code is freely available for anyone to use, modify, and distribute.
5. **Python is portable:** Code written in Python can be run on multiple platforms including Raspberry Pi OS, other Linux distributions, Windows, and macOS.

Other languages all offer advantages, and we think Python is a great place to start.

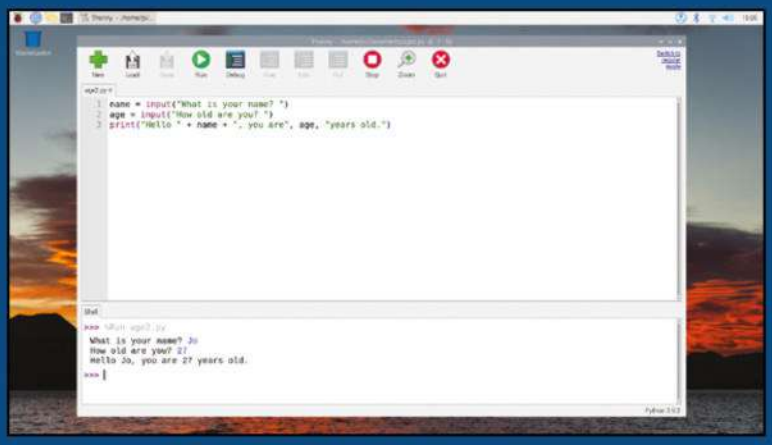
Tip!

Name your program

Python programs are typically named as all lower-case with underscores splitting multiple words, i.e.:

- **game.py**
- **hello_world.py**
- **number_generator.py**

It's a good habit to make sure your file name reflects what the program does. Try to avoid spaces and generic names.



Variables can be set to input from the user; for neatness, include a space after the text

```
print(128)
```

Click Run to print the integer 128 to the Shell. We can also use standard mathematical operators (addition, subtraction, multiplication, and division) on numbers. For example:

> We store numbers and strings in objects called variables

```
print(128+57)
```

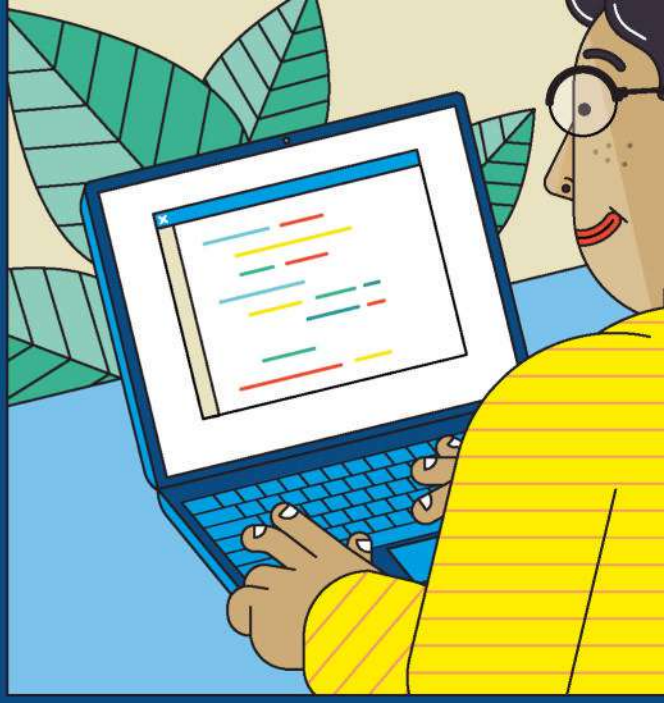
This will produce the result 185. The * and / symbols are used for multiplication and division. For example:

```
print(128/8+5*7)
```

...produces the result 51.0. There are two things to note here. First, using the divide operator automatically gives a floating-point result (hence the decimal place). Second, the BODMAS/PEMDAS rule applies, so division and multiplication are performed before addition and subtraction.

The addition operator can also be used with strings. For example:

Testing multiple conditions with if and elif statements; the following indented line only runs if the condition is true

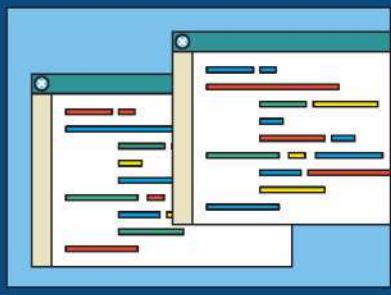


```
print("The MagPi" + " is " + "a magazine.")
```

Note the spaces around the word 'is'. Other maths operators include ** for 'to the power', % for modulo (division remainder), and // for floor division (rounded down to the nearest integer). For example:

```
print(2**4)
print(20%7)
print(22//7)
```

Which give the results 16, 6, and 3 respectively.



Variables

We store numbers or strings in objects called variables – so called because we can change (or 'vary') their value during our program.

SWITCH DATA TYPE

It's possible to switch numbers or strings between data types if needed. To switch from a number to a string, we use the `str()` function. For example:

```
x = 42
message = "The meaning of life is "
+ str(x)
print(message)
```

To switch between number types (or convert a string comprising a number), we use the `int()` and `float()` functions.

COMPARISON OPERATORS

There are six comparison operators available in Python:

Operator	Meaning
==	is equal to
!=	is not equal to
>	is greater than
<	is less than
>=	is greater or equal to
<=	is less than or equal to

Note that you need to use == for comparisons, not = (which is used to set a value).

To define a variable, we choose a name for it and use the = symbol to set its value. For example:

```
wings = 4
animal = "bee"
print("A", animal, "has", wings, "wings.")
```

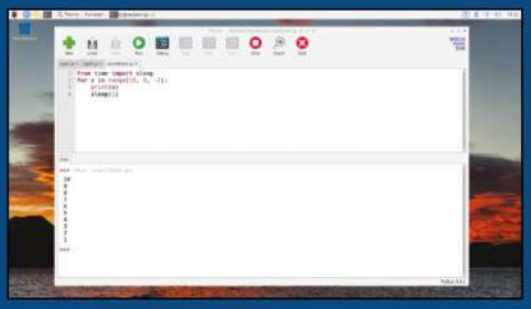
Using Python's built-in `input()` function, we can obtain user input and store it in a variable:

```
name = input("What is your name? ")
age = input("How old are you? ")
print("Hello " + name + ", you are " + age +
" years old.")
```

Conditions

A key part of many programs involves comparing variables and values to determine what step is executed next. For this, we use an `if` conditional statement.

```
age = input("How old are you? ")
age = int(age)
if age > 12 and age < 20:
    print("You are a teenager.")
```



Using a for loop to count down from ten to one, with a sleep delay between numbers

The line after the `if` statement is indented by four spaces and the indented code is executed only if the condition is true (i.e. the `age` variable is between 13 and 19). An `and` logical operator is also used in this example to check that two conditions are both true.

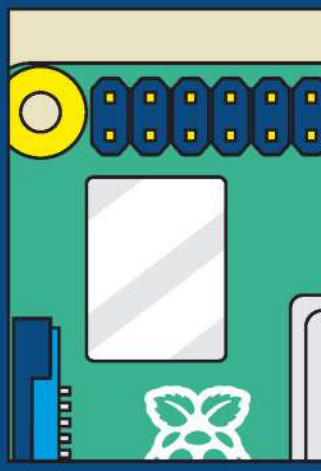
We can add an `elif` (short for 'else if') statement afterwards to perform another comparison if none of the previous ones is true:

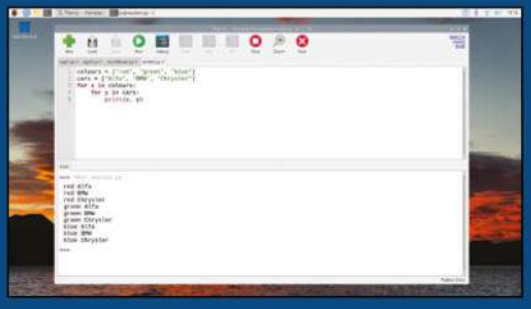
LOGICAL OPERATORS

These are used for comparing multiple conditions at once.

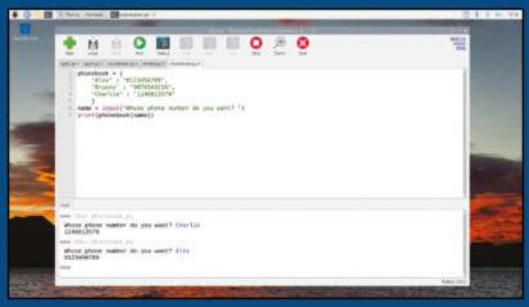
Operator	Meaning
and	both conditions are true
or	either (or both) condition is true
nor	both conditions are false

Note that the `or` operator in Python is non-exclusive, so it's still true if both conditions are true. Other operator types available in Python include identity (`is`, `is not`), membership (`in`, `not in`), and bitwise (for comparing binary numbers).

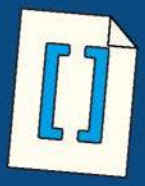




Using nested for loops to print items from two different lists



A dictionary comprises key and value pairs separated by a colon



```
elif age <= 12:
    print("You are a child.")
```

You can add as many `elif` statements as you like. At the end, you may also add an `else` statement that will only execute the next line if none of the previous conditions are true.

> A list is used to store an ordered sequence of values

```
else:
    print("You are an adult.")
```

Loops

Instead of writing lines and lines of code to perform multiple iterations, you can use a `for` loop to repeat them with different values within a range.

```
for x in range(10):
    print(x)
```

This prints the digits 0 to 9, since the default starting value for the `range()` function is zero and the 10 value we set is when it stops. To make it count from 1 to 10, we need to use:

```
for x in range(1, 11):
    print(x)
```

To count down instead, adjust the start and stop `range()` parameters and add a third one to change the step size to -1.

```
for x in range(10, 0, -1):
    print(x)
```

If you want to add a delay between printing each number, you can use the `sleep` function from Python's time module, but you need to import it first.

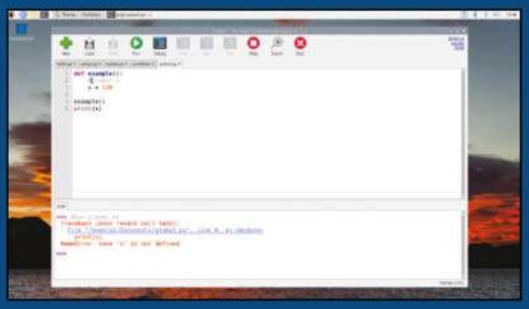
Printing out the values of tuples stored in a list

WHILE LOOP

An alternative to a `for` loop is to use `while` to test a condition and only run the loop while it is true.

```
x = 1
print("Counting from 1 to 5…")
while x <= 5:
    print(x)
    x = x +1
```

We can also use `while True:` to create a loop that runs endlessly (until you stop the program), since the condition is always true.



▲ The Shell area shows any errors when running code; here, we failed to give a function's variable global scope

```
from time import sleep
for x in range(10, 0, -1):
    print(x)
    sleep(1)
```

Another thing you can do with loops is to nest them, which we'll cover in the next section.

Lists

In Python, a list is used to store an ordered sequence of values (of any type) separated by commas. These can then be accessed individually using an index number.

```
names = ["Alex", "Brooke", "Chloe",
"David", "Ed"]
print(names[1])
```

Since Python starts counting at 0, the index of 1 (given in square brackets) selects 'Brooke' from the list.

> "A list will be used with a for loop"

Typically, a list will be used with a `for` loop, to print out a range of values.

```
fruits = ["apple", "banana", "cherry",
"damson", "elderberry"]
for i in range(5):
    print(fruits[i])
```

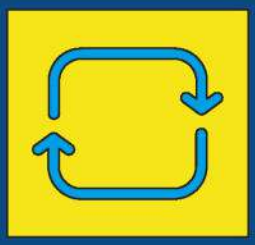
We can also nest loops to choose items from two or more lists.

```
colours = ["red", "green", "blue"]
cars = ["Alfa", "BMW", "Chrysler"]
for x in colours:
    for y in cars:
        print(x, y)
```

RETURN VALUES

Instead of using `print()` in our addition function, we could use `return` to send the result back to the caller.

```
def add(a, b):
    return a + b

result = add(138, 485)
print(result)
```



This prints nine lines, starting with 'red Alfa', 'red BMW', 'red Chrysler', 'green Alfa'…

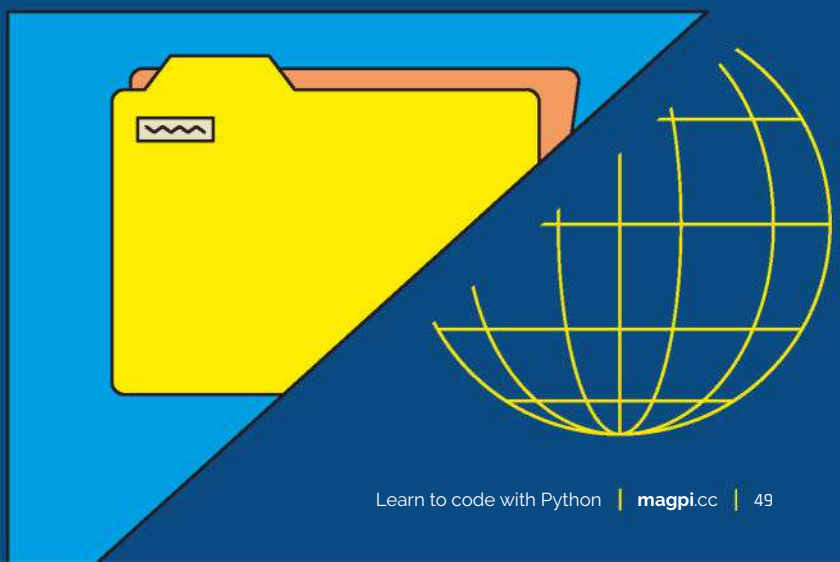
Note that here, instead of the `range()` function to set an index, we're using the `in` membership operator to select each item in turn from a list.

A list may also contain tuples. A tuple is an indexed series of comma-separated values (that can't be modified) enclosed in parentheses.

```
rgb = [(255, 0, 0), (0, 255, 0), (0, 0,
255)]
colours = ("red", "green", "blue")
for i in range(3):
    print("The RGB value for", colours[i],
"is", rgb[i])
```

Dictionaries

These are similar to lists except they contain a series of 'key:value' pairs separated by a colon, enclosed in curly braces. Instead of using an index, we use the key to access the relevant value.



VARIABLE SCOPE

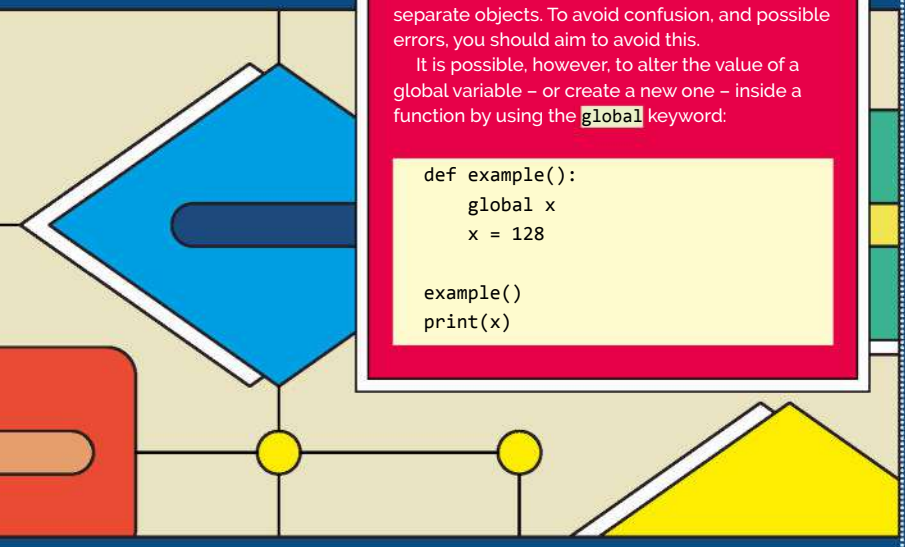
A variable created within a function is only available within that function, or any nested functions inside it. This is known as a local variable.

By contrast, a variable created within the main body of a Python program can be accessed anywhere, including in functions. This is known as a global variable.

You can even create global and local variables with the same name, as they will be treated as separate objects. To avoid confusion, and possible errors, you should aim to avoid this.

It is possible, however, to alter the value of a global variable – or create a new one – inside a function by using the `global` keyword:

```
def example():
    global x
    x = 128

example()
print(x)
```



```
phonebook = {
    "Alex" : "0123456789",
    "Bryony" : "9876543210",
    "Charlie" : "0246813579"
    }
name = input("Whose phone number do you want? ")
print(phonebook[name])
```

Note that since decimal integers can't have a leading zero in Python, we are using strings for the phone numbers.

Functions

A Python function is a block of code that only runs when 'called'. While Python has many built-in functions, you can also define your own custom ones.

Using functions in your programs is a good way to organise everything, make your code easier for others to read, and avoid having to write repetitive blocks of similar code.

To define a function, we use `def` followed by a name, parentheses (which may contain one or more arguments), and a colon. This is followed by the indented lines of code for the function. For example, a simple function to add two numbers together:

```
def add(a, b):
    result = a + b
    print("The sum is", result)
```

Now we've defined the function, we can call it with its name followed by parentheses containing the values for the arguments (if any). For example:

```
add(138, 485)
```

You could put this line in a `while True:` loop and have the user input the numbers to add.

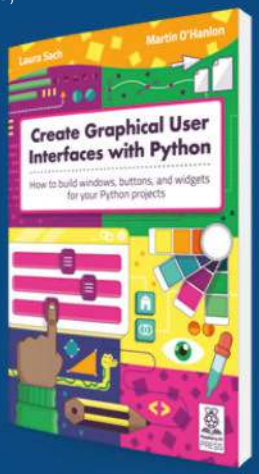
When looking at Python programs, you'll often see functions defined and called without any arguments.

```
def message():
    print("This is my message!")

message()
```

This will simply call the function to get it to execute and, in this case, print a message.

Lots more to learn

Hopefully this guide will help newcomers to get started coding with Python, but we've only really scratched the surface here. There are many more aspects to the language, such as using string formatting, classes and objects, numpy arrays, regular expressions, decorators, and handling exceptions. You can also add graphics to your programs using modules such as Pygame or Pygame Zero, and a GUI with tools such as Tkinter or guizero.


Laura Sach
Martin O'Hanlon
Create Graphical User Interfaces with Python
How to build windows, buttons, and widgets for your Python projects

Pick up a copy of *Create Graphical User Interfaces with Python* to continue your Python journey. Happy coding!

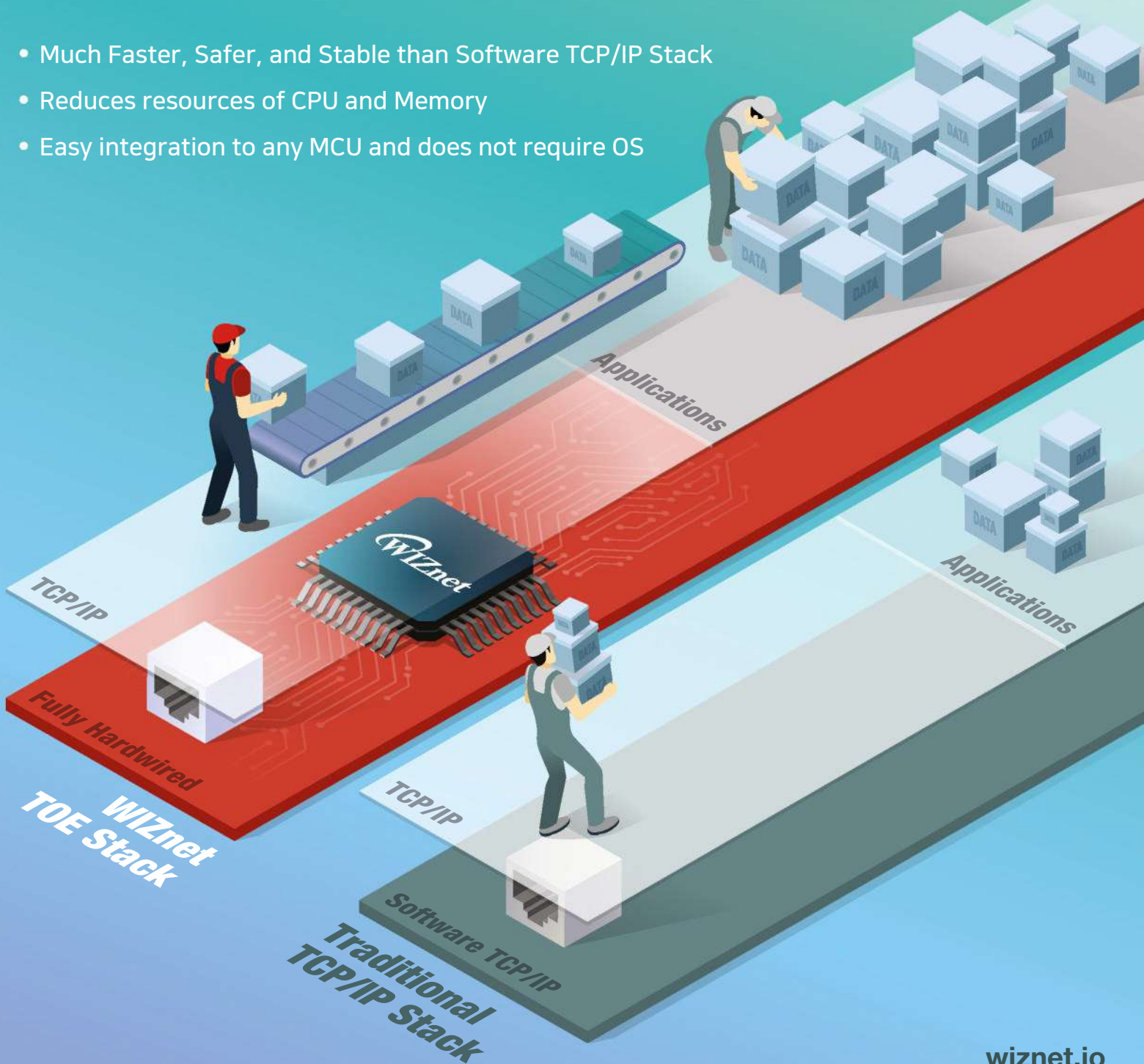
WIZnet

Internet Offload co-Processors for IoT

WIZnet TOE

TCP/IP Offload Ethernet

- Much Faster, Safer, and Stable than Software TCP/IP Stack
- Reduces resources of CPU and Memory
- Easy integration to any MCU and does not require OS



wiznet.io

Part 02

Beginning electronics: Introduction to microcontrollers

Learn about microcontrollers using Raspberry Pi Pico. Build a motion sensor which turns on LEDs whenever someone enters a room

MAKER

Stewart Watkiss

Also known as Penguin Tutor. Maker and YouTuber that loves all things Raspberry Pi and Pico. Author of Learn Electronics with Raspberry Pi.

penguintutor.com

twitter.com/stewartwatkiss

You'll Need

- Breadboard **magpi.cc/breadboardhalf**
- PIR motion sensor **magpi.cc/pir**
- White LEDs **magpi.cc/whiteled**
- 150 Ω resistors **magpi.cc/14wresistor**
- Jumper wires **magpi.cc/jumperbumper**

In this project, you will learn how to detect when someone enters a room using a PIR (passive infrared) motion sensor. You will learn how to use MicroPython to carry out actions based on the PIR sensor. The code can make decisions and set an output. This can be used to turn on LEDs. You'll also learn about schematic circuit diagrams, making it easier to understand how a circuit works.

01 What is a microcontroller?

A microcontroller is an electronic device. It is a type of computer, in that it runs code to determine what actions to take, but it isn't as powerful as your Raspberry Pi computer and does not run an operating system. The microcontroller runs your code directly and is great for using as part of an electronic circuit.

You'll need a Raspberry Pi Pico for this project. Pico uses the RP2040 microcontroller. It's a low-cost microcontroller packed full of features. There are versions of the Pico available with wireless connectivity, but the standard Raspberry Pi Pico is fine for this project.

02 Getting started with the Pico

To be able to insert the Pico onto a breadboard you'll need pins soldered to it. You can solder headers on yourself, or you can buy versions of the Pico with the pins pre-soldered. With the headers in place, you can plug the Pico into a breadboard ready for adding other components.

You'll also need to install MicroPython onto the Pico. Download the appropriate UF2 file from **magpi.cc/micropython**. Hold down the BOOTSEL button whilst connecting the Pico to your computer. Drag and drop the UF2 file to the RPI-RP2 drive that appears on your computer.

03 The PIR sensor and the GPIO pins

A PIR sensor is a device that can detect when someone is nearby. It stands for passive infrared and is essentially a heat detector, which detects the heat from someone's body. To prevent unwanted triggering, the sensor is split into zones; it only sends a high output when it detects heat moving across the zones.

The PIR sensor will be connected to one of the GPIO (general-purpose input/output) pins on

the Pico. These are sometimes referred to as GP (general-purpose) pins. The GPIO port numbers do not line up with the physical pin number. See the Pico pinout diagram (**Figure 1**, overleaf) for details.

04 Connect the PIR sensor

The Pico uses 3.3V for the GPIO pins. The PIR sensor is connected to the USB voltage which is approximately 5V, but fortunately the PIR's high output signal is only 3V, which is safe to use with the Pico.

The PIR sensor normally comes with male headers. You will need to use male-to-female jumper leads to connect from the breadboard to the PIR sensor. Connect the pin marked GND (ground) on the PIR to the breadboard row

> "The PIR sensor normally comes with male headers"

connected to pin 3 on the Pico, the VCC (power) pin to pin 40 on the Pico, and the OUT pin to pin 4 on the Pico. In the diagram (**Figure 2**), we've used the horizontal power rails of the breadboard, which makes adding additional components easier.

05 Adding LEDs

The aim in this project is to create light when the PIR sensor detects someone in the room. Ideally this will turn on a bright light, but that would need more power than the GPIO ports can provide. So, for now you can use multiple white LEDs.

One thing to be aware of is that the GPIO pins can only provide a small current. According to the RP2040 data sheet, the total across all the GPIO pins is 50mA. The highest current mode for an individual pin is 12mA. This is not the absolute maximum, but the output voltage will decrease when exceeding that current.

06 Limiting the current through the LEDs

This circuit uses four LEDs; limiting them to 12mA will be just within the maximum for the individual pins and the total for all pins. Using white LEDs,

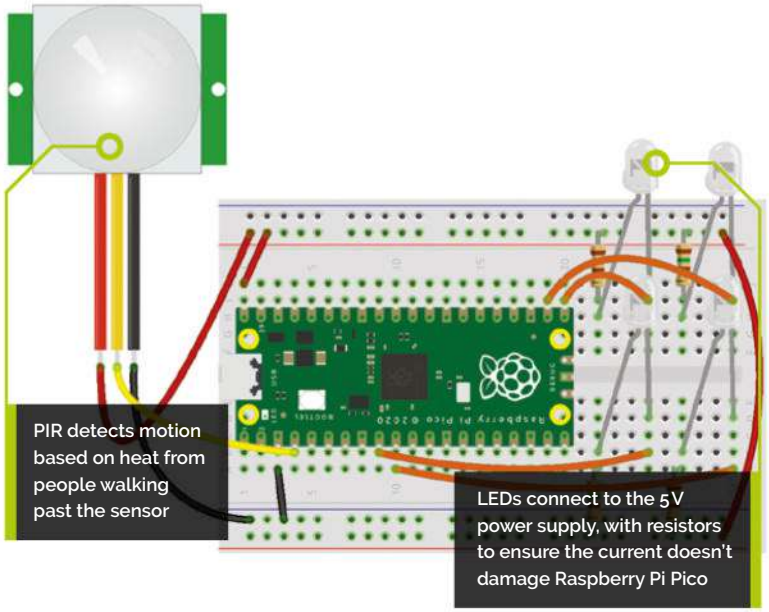

PIR detects motion based on heat from people walking past the sensor
LEDs connect to the 5V power supply, with resistors to ensure the current doesn't damage Raspberry Pi Pico

Figure 2 The breadboard wiring diagram for the project

they typically drop 3.3V across them; with a 5V power supply, then, we need a suitable resistor to limit the current to 12mA.

Using a maximum of 1.7V dropped across the resistor and limiting to 12mA gives 141Ω, which rounds up to 150Ω. These should be wired as shown in **Figure 1** over the page.

07 Schematic circuit diagram

One of the challenges with the breadboard diagram is that as more components are added, it becomes harder to see where each of the device legs and wires go.

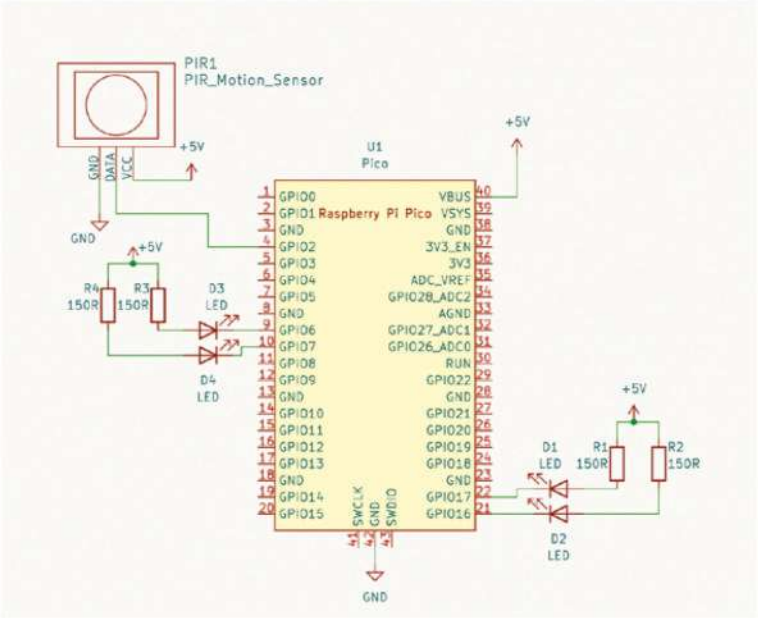


Figure 3 A schematic circuit diagram does not look the same as how it does on a breadboard, but makes it easier to see which wires connect where

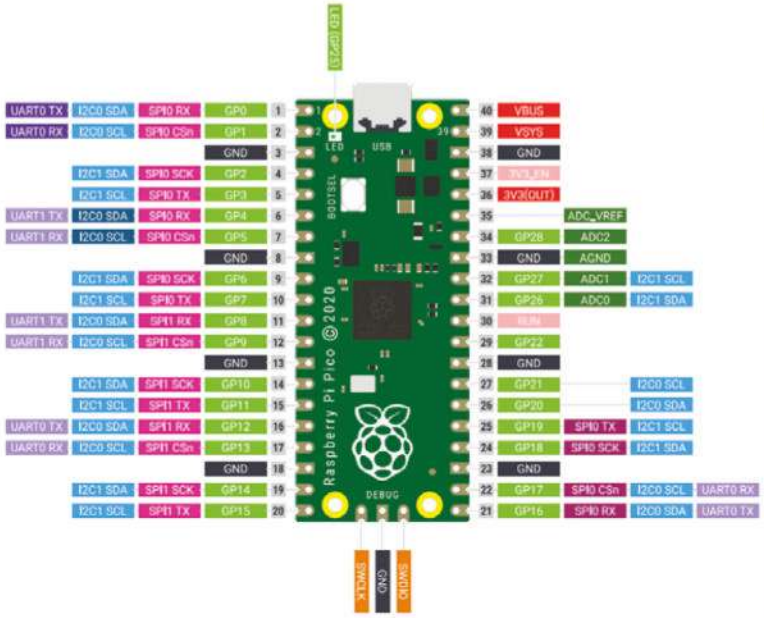


Figure 1 The location of the port numbers for the Pico are shown in this diagram. Many of these can be used for different functions

There is an alternative way of showing the circuit which is the schematic diagram. It involves learning some new symbols and conventions, but makes it much easier when creating larger circuits.

The schematic diagram for this circuit is shown in **Figure 3** on the previous page.

Top Tip

Pico GPIO is 3.3 V

The GPIO pins for the Pico are designed to work at 3.3 V, however many electronic components are designed for 5 V which can damage the Pico.

08 Understanding the schematic diagram

The schematic diagram includes a combination of boxes and other symbols which represent the components (see **Figure 4**). The components are then joined by lines representing a wire or electrical connection.

Raspberry Pi Pico is represented by the large central rectangle, which is a common way of representing integrated circuits or more complex modules. The labels around the outside represent the physical pin numbers, while the labels inside the box indicate the purpose, such as the GPIO port numbers. The PIR sensor is also a rectangle, which in this case includes an image inside to represent the sensor.

The resistors are represented by small rectangles. The LEDs are made up of a diode, which is an arrow followed by a vertical line, and then two arrows which indicate that the LED emits light. The line indicates the cathode (negative end of the diode).

Thonny - <untitled> @ 1 : 1
File Edit View
Thonny options
General
Interpreter
Editor
Theme & Font
Run & Debug
Terminal
Shell
Assistant
Which kind of interpreter should Thonny use for running your code?
MicroPython (Raspberry Pi Pico)
Details
Connect your device to the computer and select corresponding port below
(look for your device name, "USB Serial" or "UART").
If you can't find it, you may need to install proper USB driver first.
Port
Board in FS mode · Board CDC (/dev/ttyACM0)
Interrupt working program on connect
Synchronize device's real time clock
Use local time in real time clock
Restart interpreter before running a script
Install or update MicroPython
OK
Cancel
MicroPython (Raspberry Pi Pico) · /dev/ttyACM0

Select the Pico and port from the Configure Interpreter menu option. This is how the Pico appears when connected to a Raspberry Pi

09 Power connections

The power supply connections are shown through symbols. The arrows pointing upwards are used to indicate the positive power connection which is connected to 5 V from the Pico. These are all connected.

The ground, or negative, connection is shown as a small diagram with the label GND. They are all connected.

From the diagram, you can see the pins that are used on the Pico and how all of the components are wired together. For complex circuits, this is much easier than trying to follow the wires on the breadboard diagram.

10 Start programming

To program Raspberry Pi Pico, you'll first need to connect using Thonny. It may connect automatically. If not, then from the Run menu choose 'Select interpreter' and choose 'MicroPython (Raspberry Pi Pico)'. Select the detected port; if running Thonny on a Raspberry Pi, it'll normally say 'Board in FS mode'; for Windows, it'll be the appropriate USB serial port.

You can write the code in the main window and save it to the Pico. If you name the file **main.py** then it will run automatically when your Pico is powered on – any other file name and you will need to run it manually using the Thonny editor.

11 Control through code

The code starts by importing the relevant modules – the machine module is a standard MicroPython library for handling GPIO on microcontrollers, and the Pin class is used to represent an individual GPIO pin. The utime module is the MicroPython version of the standard Python time module. It's used to add delays in the code.

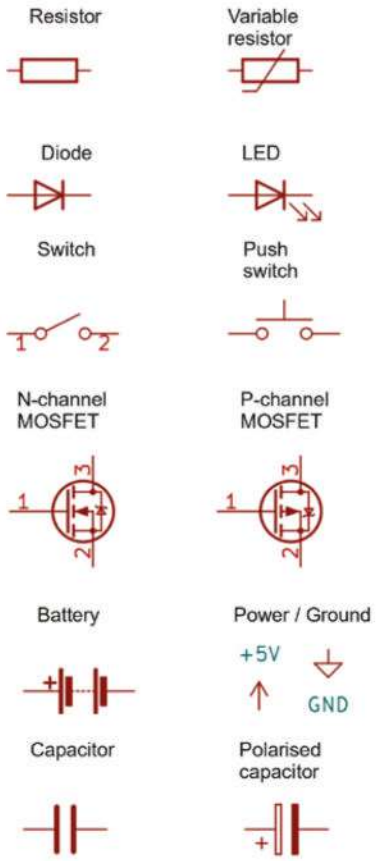


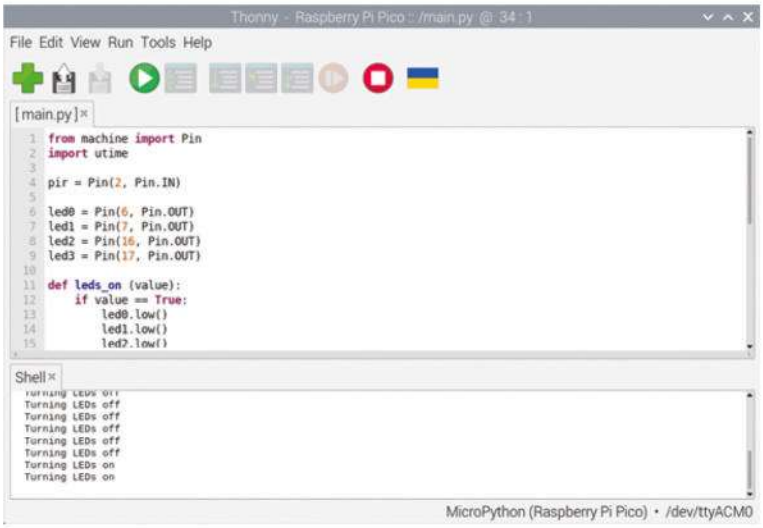
Figure 4 Some common circuit symbols used on schematic diagrams. There are different standards – these are based on symbols used by the KiCad circuit designer

The PIR sensor is used as an input and the four pins for the LEDs are defined as outputs.

There is a function called `leds_on()` which can be passed either True or False as an argument. If True is passed, then it turns all the LEDs on; if False is passed, then it turns all the LEDs off. Note that the pin is set to low to turn an LED on, and high to turn it off. This may appear counter-intuitive, but is because the LEDs are connected to the positive supply and the GPIO needs to be low to allow the current to flow into Pico.

12 Detecting the PIR value

The rest of the code is contained within a `while True:` loop, which will constantly repeat. If the PIR gives a high signal to the pin, the value is set to 1 and the LEDs turn on. If it has a low signal, then the value is 0. There is a delay in each of the conditions – it is intentionally longer when the LEDs are turned on to ensure it stays on for a reasonable period of time when triggered. In this case it stays on for ten seconds, but if the PIR does not detect anyone, then it checks every second.



The Thonny editor can be used to write MicroPython code directly onto the Pico. Switch to standard mode to see all the options

main.py

DOWNLOAD THE FULL CODE: magpi.cc/picopir

Language: **MicroPython**

```
001. from machine import Pin
002. import utime
003.
004. pir = Pin(2, Pin.IN)
005.
006. led0 = Pin(6, Pin.OUT)
007. led1 = Pin(7, Pin.OUT)
008. led2 = Pin(16, Pin.OUT)
009. led3 = Pin(17, Pin.OUT)
010.
011. def leds_on (value):
012.     if value == True:
013.         led0.low()
014.         led1.low()
015.         led2.low()
016.         led3.low()
017.     else:
018.         led0.high()
019.         led1.high()
020.         led2.high()
021.         led3.high()
022.
023. while True:
024.     if (pir.value() == 1):
025.         print ("Turning LEDs on")
026.         leds_on(True)
027.         utime.sleep (10)
028.     else:
029.         print ("Turning LEDs off")
030.         leds_on(False)
031.         utime.sleep (1)
```

Top Tip

Pico numbering

With the USB connector positioned to the left, the pins are numbered starting at 1 in the bottom left. Along the bottom row up to pin 20, then from right to left along the top row from 21 to 40.

Machine learning-powered private speech transcriber

Keep your personal data private by carrying out automated voice transcription on your own Raspberry Pi

MAKER

K.G. Orphanides

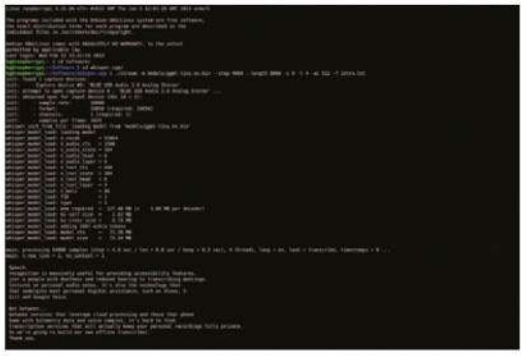
KG is a writer, developer and software preservationist. They're sceptical of the AI-in-everything fad, but also don't want a human transcriber to hear their audio diary.

magpi.cc/hauntedowlbear

Speech recognition is massively useful, from providing accessibility features to people with deafness and reduced hearing to transcribing meetings, lectures, or personal audio notes. It's also the technology that underpins most personal digital assistants.

But between services that leverage cloud processing and those that phone home with telemetry data and voice samples for later analysis by humans, it's hard to find transcription services that keep your personal recordings fully private. So we're going to build our own offline transcriber.

Whisper.cpp isn't perfect. It can't distinguish between different speakers and, like most transcribers, it can get confused by cross-talk and indistinct enunciation. However, its performance rivals that of many commercial services, and your data never leaves your hands.



Although less accurate than 'main', if want to make a hands-free note or have the text of a meeting immediately available, you'll find the near-real-time 'steam' transcription tool useful

You'll Need

- Georgi Gerganov's Whisper.cpp port **magpi.cc/whispercpp**
- A microphone (for live transcription)

01 Prepare your system

Machine learning applications will cheerfully use as much processor power as you can throw at them – during cross-platform testing, we maxed out all 16 threads of a hefty AMD Zen CPU. Although Georgi Gerganov's Whisper port is optimised for efficiency, you can expect better results from more powerful systems. We used a Raspberry Pi 400 with 4GB RAM. Unless you just want to transcribe audio files, you'll also need a microphone – we used a Blue Yeti USB mic. For somewhat better performance, you could use an 8GB Raspberry Pi 4 running Raspberry Pi OS Lite to save on the overheads of the GUI.

02 Use the source

There are no pre-compiled binaries for ARM platforms, so you'll either have to clone Whisper.cpp from its Git repository, or download a release version of the source code from **magpi.cc/whisperreleases**. We've gone with

the former, but when in doubt, grabbing source code from the Releases page is always a safe bet. The current version of the master repository will always have the latest features, but there's a chance that you'll get the latest bugs, too.

```
sudo apt install libsdl2-dev
git clone https://github.com/ggerganov/whisper.cpp
cd whisper.cpp
```

Unless specifically stated otherwise, all commands in this tutorial are run from this directory (**whisper.cpp**).

03 The model

We're going to use the recommended tiny Whisper model for Raspberry Pi, which requires just 75MB of disk space and around 125MB RAM. While, on paper, the larger and theoretically more accurate base model might be assumed to work on Raspberry Pi, in practice the processing overhead resulted in breaks and delays in transcription, so we're going to keep it small.

```
./models/download-ggml-model.sh tiny.en
make -j tiny.en
make -j stream
```

> "There are no pre-compiled binaries for ARM platforms"

When the build info is shown after making stream, you'll notice that the makefile automatically optimises for armv71, including options such as neon-fp-armv8.

04 Take dictation

The stream tool is your live transcriber. Everything you say into the mic will be passed through Whisper and displayed on screen. At the terminal, type:

```
./stream -m models/ggml-tiny.en.bin --step 4000 --length 8000 -c 0 -t 4 -ac 512
```

By using the smallest Whisper model and tweaking its listening settings, the 'main' file transcription tool can transcribe, display, and export text based on a WAV file
The --print-colors option will show you how much confidence Whisper has in the transcription of each word

This command optimises for live transcription performance on Raspberry Pi's modest hardware by sampling the audio every four seconds with the `--step 4000` instruction. One second would be `--step 1000`.

05 Transcribe to file

Displaying text on-screen after a short delay is undeniably useful, particularly if you have difficulty hearing or following spoken lectures or conversations. However, you'll usually want to keep the transcribed text for future use. To do this, append `-f filename.txt` to the end of the command.

Press **CTRL+C** to end transcription.

06 Ride the wave

You can also use Whisper.cpp to transcribe previously recorded files, making it a useful alternative to online AI transcription tools such as Temi and Otter.ai, with the benefit of knowing that your recording never leaves your hands.

Top Tip

Use the best mic you can

Microphone quality makes a huge difference to the accuracy of Whisper.cpp's live audio transcription.

GNU nano 5.4
gb0.wa
Good morning. This Tuesday is Election Day. After months of
spirited debate in vigorous campaigning, the time has come
for Americans to make important decisions about our nation's
future and encourage all Americans to go to the polls and vote.
Election season brings out the spirit of competition between
our political parties. And that competition is an essential part
of a healthy democracy. But as the campaigns come to a close,
Republicans, Democrats and independents can find common ground on
at least one point. Our system of representative democracy is
one of America's greatest strengths. The United States was
founded on the belief that all men are created equal. Every
election day, millions of Americans of all races, religions
and backgrounds step into voting booths throughout the nation,
whether they are richer, poor, older, young. Each of them has an
equal share in choosing the path that our country will take.
And every ballot they cast is a reminder that our founding
principles are alive and well. Voting is one of the great
privileges of American citizenship. And it is always
required brave defenders. As you head to the polls next week,
remember the sacrifices that had been made by generations of
Americans in uniform to preserve our way of life. From
Bacher Hill to Baghdad, the men and women of American armed
forces have been devoted guardians of our democracy. All of
us owe them and their families a special debt of gratitude on
the election day. Americans should also remember the important
example that our elections set throughout the world. Young
democracies from Georgia and Ukraine to Afghanistan and Iraq
can look to the United States for proof that self-government
can endure and nations that still live under tyranny and
oppression can find hope and inspiration in our commitment
to liberty. For more than two centuries, Americans have
demonstrated the ability of free people to choose their own
leaders. Our nation has flourished because of its commitment to
trusting the wisdom of our citizenry. In this year's election, we
will see this tradition continue. And we will be reminded
once again that we are blessed to live in a free nation guided
by the will of the people. Thank you for listening.

You can have the transcriber output an audio file to text, making it easy to do further work on

Currently, if you want to transcribe an audio file, it has to be in 16-bit PCM WAV format. If your recorder outputs MP3 files, or even PCM files at a different bit depth, you'll need to convert them first. We can use ffmpeg for that at the command line.

```
ffmpeg -i input.mp3 -ar 16000 -ac 1 -c:a
pcm_s16le output.wav
```

If you've got multiple MP3 files to convert, you put them all in the same directory and run:

```
for f in *.mp3; do ffmpeg -i "${f}" -ar
16000 -ac 1 -c:a pcm_s16le  "${f%.*}.wav" ;
done
```

Top Tip

Multilingualism?

Although OpenAI's Whisper was trained on multilingual speech data, the tiny sets we're working with in this tutorial only recognised English in our tests.

07 File to text

To transcribe a pre-recorded file, we'll use the `main` command. First, let's transcribe the supplied sample **jfk.wav** file. Type:

```
==============
main: Speech detected! Processing ...
main: Heard 'Stop calling me a robot.', (t = 10616 ms)
==============
prompt:
This is a dialogue between robot (A) and a person (B). The dialogue so far is:

B: Hello robot, how are you?
A: I'm fine, thank you.
B: You're a robot, correct?
A: Hello robot, I am A, the robot who was sent to Earth to rescue me.
B: How did you get to earth?
A: Hello robot, how are you?
B: Do you have a spaceship?
A: Hello robot, how are you?
B: I'm not a robot, you are.
A: Hello robot, how are you?
B: Stop calling me a robot.
A: Hello robot, how are you?

Here is how robot (A) continues the dialogue:
```

There's even a two-way voice chatbot function in here, but don't expect the best results with the smallest GTP-2 models available

```
./main -m models/ggml-tiny.en.bin samples/
jfk.wav --output-txt
```

The output text file will be placed in the same directory as the WAV file you transcribed. You can also transcribe to CSV format with time stamps using the `--output-csv` option.

To see words in the on-screen stream (but not the exported transcript file) colour-coded by confidence, include `--print-colors`, thus:

```
./main -m models/ggml-tiny.en.bin --print-
colors samples/jfk.wav --output-txt
```

You can have it transcribe multiple files using wild cards:

```
./main -m models/ggml-tiny.en.bin *.wav
--output-txt
```

> "It's rich ground for experimentation using different models"

Or by giving it multiple file names to work with:

```
./main -m models/ggml-tiny.en.bin file1.wav
file2.wav --output-txt
```

If you want some more audio samples to play with, type:

```
make samples
```

08 Moving the conversation on

For the purposes of this feature, we're interested in privacy-focused transcription, but Whisper.cpp's examples include support for GPT-2 conversational AI, so we might as well implement this feature as a novelty.

At the time of writing, the instructions for obtaining a pre-generated ggml Tensor library compatible GTP-2 model are incorrect. Functional versions of Gerganov's converted small GTP-2 model can be found at **ggml.ggerganov.com**

as well as **magpi.cc/ggml**. Check these if our instructions go out of date due to file location changes, or see **magpi.cc/gpt2** for instructions on manually converting TensorFlow models.

09 Summoning Santa

We're going to spin up Whisper.cpp's integrated AI chatbot and install the espeak speech synthesiser, which is the default option for voicing the bot's speech. In Whisper's main directory, at the terminal, type:

```
wget --quiet --show-progress -O models/
ggml-gpt-2-117M.bin https://huggingface.co/
datasets/ggerganov/ggml/blob/main/ggml-model-
gpt-2-117M.bin
sudo apt install espeak
make talk
./talk -mw models/ggml-tiny.en.bin -p Santa
```

This demo throws you into a conversation with Santa, once you've provided an extra prompt. It's a bit slow on Raspberry Pi and the conversation wasn't very satisfying, or believable, on any system we tested it on. It's rich ground for experimentation using different models and prompts, but this mostly requires more processor power than your average SBC has to offer. You can change the identity of your interlocutor by following the `-p` switch with a different name.

10 Extra features

Also included in Whisper.cpp's **examples** directory is **generate-karaoke.sh** which, once you've done `sudo apt install sox` to get an audio processor, will attempt to generate a karaoke video based on audio input from the microphone.

More usefully, there's a benchmarking tool to help you compare performance between your devices, a livestream audio transcription tool, and a YouTube-oriented video download and subtitling tool, which has yt-dlp as a dependency. The transcription is added to the downloaded video as a subtitle file, and although it's not great with non-Anglo accents, it's actually an improvement on YouTube's automated subtitling. Note that all of these use the ggml-base.en.bin model by default, so you'll need to change the model paths in their .sh files to use the tiny model that's best optimised for Raspberry Pi.

The subtitling tool often does a better job than YouTube's integration closed captioner, but it's not optimised for Raspberry Pi by default

11 Further ideas: Command station

If you're hoping to use Whisper.cpp to build your own offline digital voice assistant, there's a reference example called 'command' to get you started. First, in the **whisper.cpp** directory:

```
make command
./command -m ./models/ggml-tiny.en.bin -ac
768 -t 3 -c 0
```

This launch string is optimised for Raspberry Pi. Once launched, command activates with the spoken instruction 'OK Whisper, start listening for commands' and specifically listens for short trigger phrases. Note that these aren't actually assigned to any actions or responses in this sample tool.

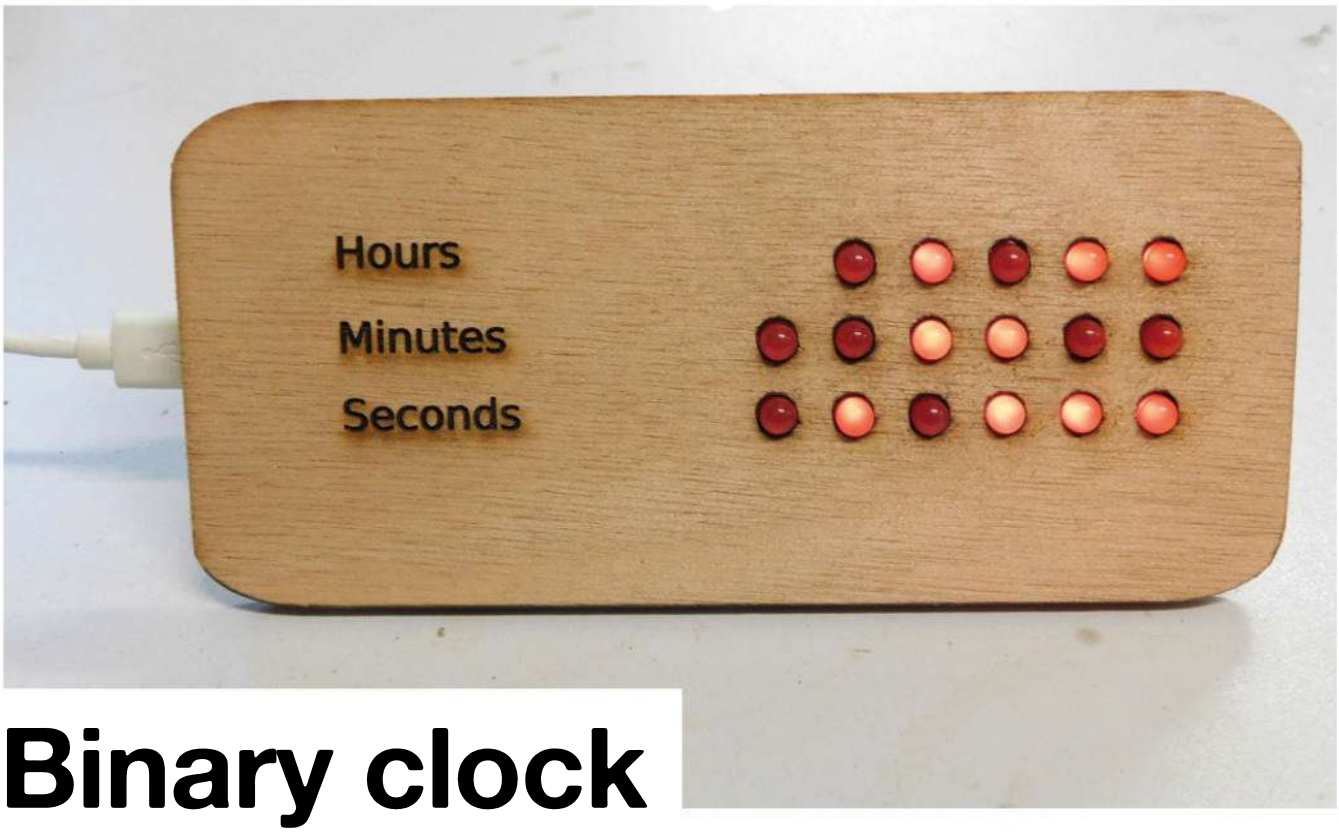
We found that this short phrase detection behaviour was more susceptible to interference from background noise than stream's continuous real-time transcription, but if you're after a framework to build your own voice automation system from, this is certainly a good place to start.

Working with Whisper

OpenAI's Whisper neural net was trained on 680,000 hours of speech data scraped from the web, covering multiple languages. This gives it remarkable accuracy, Although it's not free of worrying ethical implications about both whether that speech data should have been used like this and how Whisper itself could be used, the nature of speech recognition means that at least we won't be dealing with the problems of creative plagiarism that affect generative AIs designed to 'create' text and images.

Spend enough time playing with it, particularly if it doesn't clearly catch someone's words, and you'll see Whisper inferring content that has nothing to do with what you actually said, but does have a kind of internal, albeit fictitious logic. It's this kind of behaviour that leads people prone to anthropomorphising technology to falsely ascribe sentience to machine learning tools.

Our favourite example generated from entirely unrelated words and a somewhat shoddy microphone was: 'Thanks, both of the zombies!' 'Thank you, Master, we love you.'



Binary clock

Keep track of time, and learn to count in 1s and 0s

Ben Everard

@ben_everard

Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.

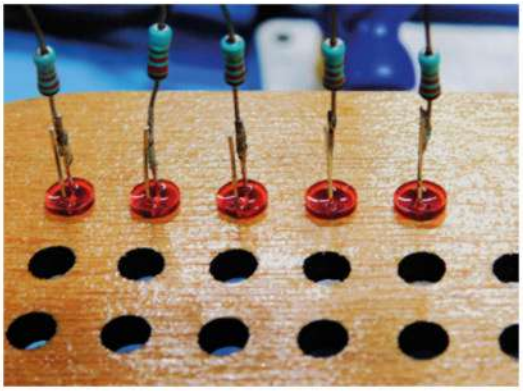
Binary is the key concept in how computers operate. If you drill down far enough on any operation, transmission, or storage, you'll find 1s and 0s, on or off, positive or negative. This ability to build up to complex behaviours from a simple distinction between, and manipulation of two states, is what computing is all about.

Yet, despite this being absolutely fundamental to the way computers work, it can be a bit alien to us humans. We're brought up in a world of ten digits, not just two, so it can be hard to get an intuitive sense of binary, especially if you don't use it very much. Our solution is to make a clock that displays time in binary.

We're going to use MicroPython for this because it's quick and easy to get started wirth, and has access to Pico's timekeeping features. While MicroPython is fairly standard across several boards, this particular project requires having an on-board real-time clock (RTC), so won't work with all boards. We're going to be using a Pico W, but it might work on others that have both wireless networking and RTC hardware. Along the way, we'll learn how to connect Pico W to the internet to get the time, and how to use Pico's Real Time Counter to keep track of the time in hours, minutes, and seconds. Let's take a look at this first.

There's a protocol for getting time on a network, and it's unimaginatively named Network Time Protocol (NTP). This actually does a few things around synchronising time over a network with various degrees of lag, but we don't need to worry about being a few milliseconds out, so we're just going to grab the current time.

We're also not going to worry about daylight savings time. We're doing all this in MicroPython, so we're going to take the simplest solution – twice a



Left
Soldering resistors in-line is an easy way to build circuits without a circuit board

year, you'll connect to your clock and adjust the offset manually. If you want, you could add a toggle switch that flips back and forwards to adjust for daylight savings.

We suspected that setting the time on an RTC via NTP might be a popular thing to do, so we wrapped up the example code for NTP in a simple library that does just this – you can get it from **hsmag.cc/SimpleNTP**.

You'll need to copy the **simple_ntp.py** file over to your device – you can do this by opening it in Thonny, then selecting 'Save as' and MicroPython device. Remember, it must have the same name on the device.

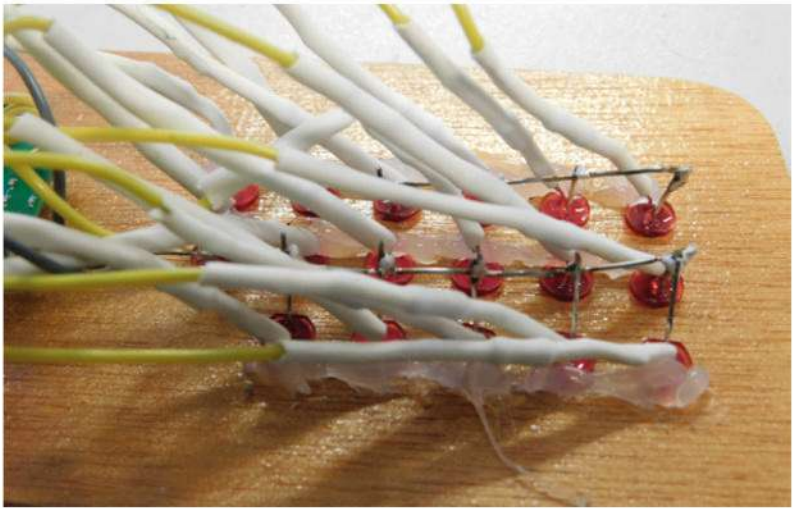
We'll shortly look at what's in that file, but for now, we'll use it to grab the time and set the RTC. You can do this with the following code:

```
import network
import socket
import time
import struct
from machine import Pin
import simple_ntp

led = Pin("LED", Pin.OUT)

ssid = 'your_ssid'
password = 'your_password'


wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect(ssid, password)

max_wait = 10
while max_wait > 0:
    if wlan.status() < 0 or wlan.status() >= 3:
        break
    max_wait -= 1
    print('waiting for connection...')
    time.sleep(1)

if wlan.status() != 3:
    raise RuntimeError('network connection
failed')
else:
    print('connected')
    status = wlan.ifconfig()
    print( 'ip = ' + status[0] )

led.on()
simple_ntp.set_time()
print(time.localtime())
led.off()
```

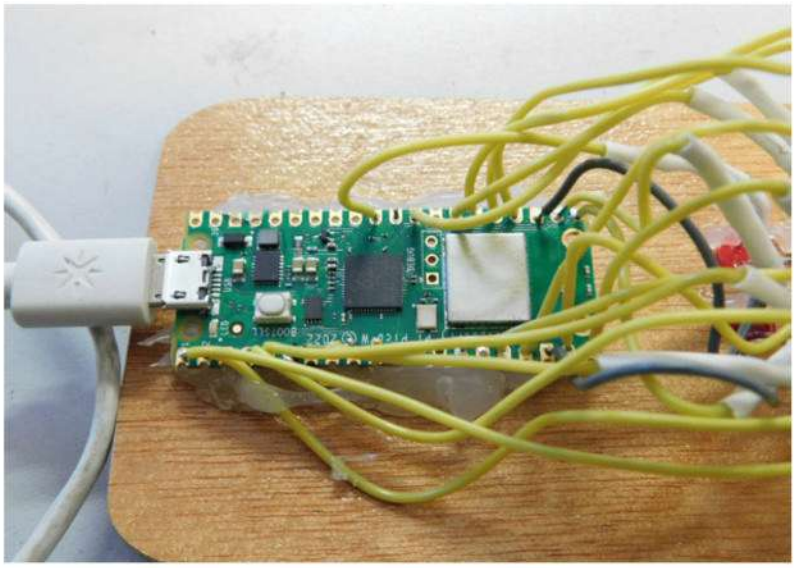
Obviously, you'll need to put in your network's SSID and password, but other than that, it should just run. At the end of it, it'll print the result of `time.localtime()`. Firstly, we should note that this is a misleading name as it won't print your local time, it'll print the time in GMT. Secondly, it won't actually print the time in a usual format, it'll print it as a list. The items are: [year, month, monthday, hour, minute, second, weekday, yearday]. All of these should be set correctly.

DIMMING

Our program turns the LEDs all the way on or all the way off. This is fine for us, and we can control the brightness by picking the right resistor. However, you can control the brightness of LEDs by flicking them on and off really fast. This is known as pulse width modulation (PWM). Pico W does support PWM but is limited to just 16 pins, so we can't use it in this project. We could create a PIO program that took in details of all the pins that should be on, and do a global PWM across all of them, but this is more than we need.



Below
The ground wires connect across all the LEDs



Above
Pico W is a great choice for this because it's cheap, has a lot of GPIOs, has WiFi for time, and has an RTC

This uses `simple_ntp.set_time()` to set Pico W's RTC to the correct time. We won't go through the whole of this, but let's look at a few key bits. The function is defined with:

```
def set_time(offset=0, delta=2208988800,
host="pool.ntp.org"):
```

There are three optional parameters here. The first is `offset`, which is a number in seconds that you want to set your RTC's time to – this is useful if you want to set it to a particular time zone. The second is `delta`, which you can probably ignore most of the time, but it's the difference between the way time is encoded in NTP. The final parameter is the NTP server. Again, `pool.ntp.org` is probably a good choice for the vast majority of cases. This isn't a specific server but a domain that will resolve to an NTP server close to your location. By being close to you, it should (in theory at least) have a low latency and, therefore, you'll get a more accurate time. However, for our purposes, the difference between high- and low-latency servers should be irrelevant.

Now we've got our time, we will need to create the hardware to display it. Here, it's up to you what you want to do. We obviously need a lot of LEDs, but how you mount them depends on what you have and what look you want to create. You could create a bare-bones look on perfboard. You could design a PCB for it. You could get some wood and drill holes to mount the LEDs in. Or, you could 3D-print a front panel for it. We opted to laser-cut ours because we wanted a wooden front panel, but any of these options should be reasonably straightforward.

Our laser-cut panel has holes for 5 mm LEDs in a line, so you can push the LEDs in place and secure them with a drop of hot glue.

The circuit is quite simple because the output is in binary, and our GPIOs work in binary. Each GPIO connects to a current-limiting resistor, then to the positive leg of an LED (usually the longer leg), and the negative leg of the resistor connects to ground. There are more LEDs than ground connections available, so you'll need to connect them all together.

The resistors need to be at least 220 Ω, but you might want to make them higher to limit the brightness of the LEDs. This will depend a lot on the particular LEDs you have, and how bright you want them. We found that around 1 kΩ usually works well, but it's obviously worth experimenting to see what you like.

> We've used the classic 5 mm red LEDs that give a retro feel, **but 3 mm LEDs will give you a more compact design**

You can use any LEDs that work with 3.3 V. They can be any colour and any form factor. We've used the classic 5 mm red LEDs that give a retro feel, but 3 mm LEDs will give you a more compact design. You could use surface-mount LEDs if you're making it on a PCB. You could even get creative and use 3 V flex LED 'noodles' to build a more abstract-looking clock.

We need five pins to display the hours (in 24-hour format) and six each for minutes and seconds. That means 17 LEDs in total on the GPIO pins 0 to 16.

We found it easiest to first solder the LEDs and resistors together, then put the LEDs in their holes, and then solder the connectors to Pico W's GPIO pins and ground. One thing you need to be aware of is making sure that there's no risk of short circuits if anything gets bent. We did this by covering the LED positive leg and resistor in heat shrink. That way, all the ground wires are exposed, but it doesn't matter if anything shorts between them. It would perhaps be a bit more secure if the grounds were also protected – it's up to you how you set it up. If you don't have heat

NEW TO MICROPYTHON?

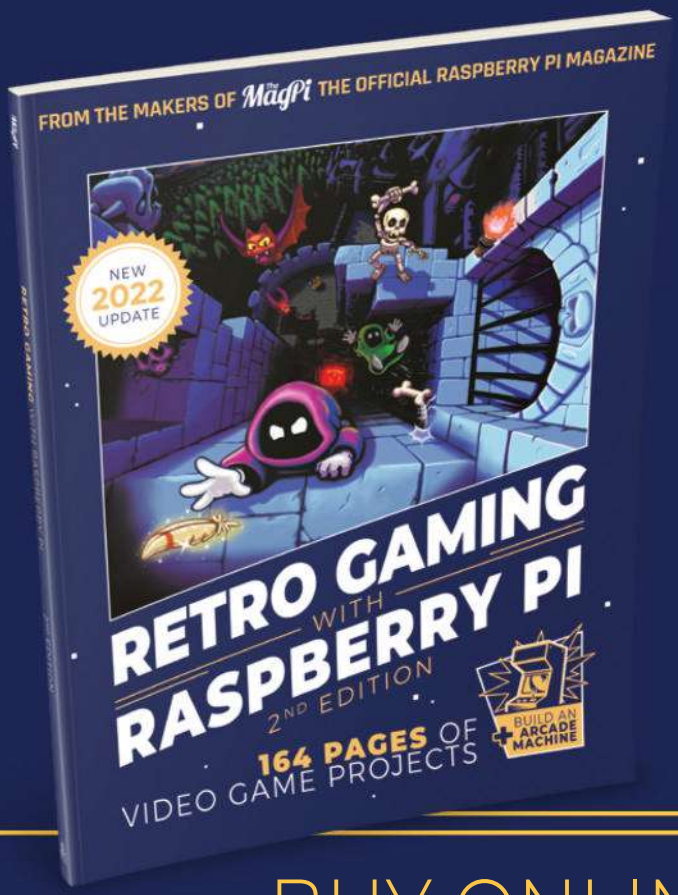
If you've not used MicroPython before, you have an exciting time ahead of you! It's a great way of programming Pico and Pico W with very little setup. It's powerful, and user-friendly. There's a quick guide to getting started at **hsmag.cc/DocMicroPyth**. For a more in-depth guide, take a look at our book *Get Started with MicroPython on Raspberry Pi Pico*, which you can download for free or buy in print at **hsmag.cc/mpbook**.

RETRO GAMING WITH RASPBERRY PI

2ND EDITION

Retro Gaming with Raspberry Pi shows you how to set up a Raspberry Pi to play classic games. Build your own games console or full-size arcade cabinet, install emulation software and download classic arcade games with our step-by-step guides. Want to make games? Learn how to code your own with Python and Pygame Zero.

- *Set up Raspberry Pi for retro gaming*
- *Emulate classic computers and consoles*
- *Learn to code your own retro-style games*
- *Build a console, handheld, and full-size arcade machine*


FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE
NEW 2022 UPDATE
RETRO GAMING
WITH
RASPBERRY PI
2ND EDITION
164 PAGES OF VIDEO GAME PROJECTS
+ BUILD AN ARCADE MACHINE

BUY.ONLINE:

magpi.cc/store

English not your mother tongue?

The MagPi is also available in German!

Subscribe to the German edition of The MagPi and get a Raspberry Pi Pico with headers and a cool welcome box **FOR FREE!**

Use the coupon code 115PicoDE on www.magpi.de/115

shrink, you could also cover it with electrical tape, but given how much there is to do, it might be easier to just get some heat shrink.

Now we've done the hardware, it's time to turn our attention back to the software. The main thing we need to do is convert the time as numbers into the LEDs that we want to turn on and off. We've done this in a method called **display_num**. This method takes two arguments: the number you want to display, and a list of pins on which to display it.

Converting a number to binary isn't too hard. We've used **enumerate(pins)**. This is a useful Python method when you want to loop through an iterable but still want a counter – as you can see, it returns both.

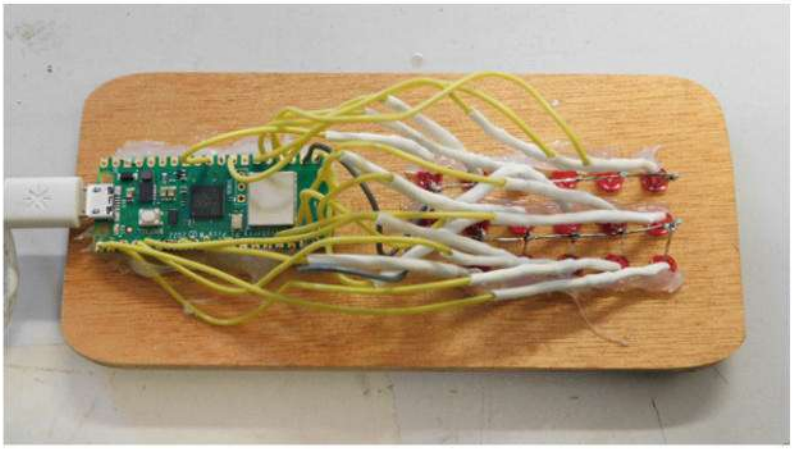
Each loop, we want to know if the number remaining is greater than or equal to the current digit. If it is, we light that digit up and take the value of that digit away from the number, if it's not, we turn that LED off and move to the next number.

Since Pico W's RTC can keep track of time, we can just grab the time when the computer boots up and let Pico W take care of the rest.

Here's our final code for the clock:

```
import network
import socket
import time
import struct
from machine import Pin
import simple_ntp

led = Pin("LED", Pin.OUT)

ssid = 'your_ssid'
password = 'your_password'

hour_pins =  [Pin(4,Pin.OUT), Pin(3,Pin.OUT),
Pin(2,Pin.OUT), Pin(1,Pin.OUT), Pin(0,Pin.OUT)]
minute_pins = [Pin(17,Pin.OUT),Pin(16,Pin.
OUT),Pin(15,Pin.OUT),Pin(14,Pin.OUT),Pin(13,Pin.
OUT),Pin(12,Pin.OUT)]
second_pins = [Pin(26,Pin.OUT), Pin(22,Pin.
OUT),Pin(21,Pin.OUT), Pin(20,Pin.OUT),Pin(19,Pin.
OUT),Pin(18,Pin.OUT)]

wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect(ssid, password)

max_wait = 10
while max_wait > 0:
    if wlan.status() < 0 or wlan.status() >= 3:
        break
    max_wait -= 1
    print('waiting for connection...')
    time.sleep(1)

if wlan.status() != 3:
    raise RuntimeError('network connection
failed')
else:
    print('connected')
    status = wlan.ifconfig()
    print( 'ip = ' + status[0] )

led.on()
simple_ntp.set_time()
print(time.localtime())
led.off()

def display_num(number, pins):
    for counter, pin in enumerate(pins):
        if number >= pow(2, len(pins)-
(counter+1)):
            number = number - pow(2, len(pins)-
(counter+1))
            pin.value(1)
        else:
            pin.value(0)

while True:
    display_num(time.localtime()[3], hour_pins)
    display_num(time.localtime()[4], minute_pins)
    display_num(time.localtime()[5], second_pins)
    time.sleep(0.1)
```

This is a very bare-bones clock, but you can extend this basic timepiece in any way you like. You could add an alarm, a date function, a time control, or anything else you'd like to keep track of.



Below
We attached Pico W with hot glue, but you could also use screws in the mounting holes

HackSpace

This tutorial is from HackSpace magazine. Each issue includes a huge variety of maker projects inside and outside of the sphere of Raspberry Pi, and also has amazing tutorials. Find out more at **hsmag.cc**.

ASTROPHOTOGRAPHY with Raspberry Pi

Get stunning shots of the cosmos with your Raspberry Pi, a Camera Module, and some smart code. By stellar guide **Rob Zwetsloot**

Looking up at the night sky is something we've been doing since ancient times, with stars, planets, and moons shining down from the cosmos to remind us that there's a whole universe beyond Earth. Taking photos of the night sky to capture these sights can be a little tricky, though.

This is where Raspberry Pi comes in. With a range of excellent Camera Modules and customisable automation, you can pretty easily create a setup that will catch the wonders of the night and the beauty of far-off celestial bodies. It's time to heed the call of the night.

Basic sky photography

Get the best cosmic shots with these out-of-this-world tips

You can still get great shots without a telescope, using just a Raspberry Pi HQ Camera, if you choose the right lenses and program the shot in a specific way.

01 Choosing a lens

Without a telescope, you'll probably be aiming for night sky photos that include a lot of the sky and don't focus on celestial bodies. In this case, 16–24 mm wide-angle lenses are a good option, which means the 16 mm lens often sold with High Quality Cameras is a perfect place to start.

02 Exposure settings

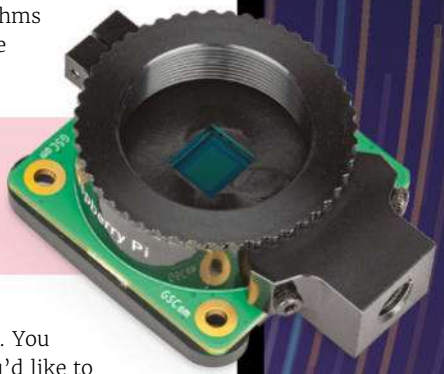
A longer exposure setting allows for more light to be captured by the sensor. For astrophotography, you normally want **15 to 30 seconds** of exposure so enough light data can be captured. The libcamera library is programmed in microseconds, so this would be a value of 15,000,000 and 30,000,000 respectively.

03 Command-line code

With the right lens and knowledge of exposure settings, we can start building a command that works in libcamera. For long exposures, we need to disable some automatic control algorithms for the best results. Overall, the command looks like this:

```
libcamera-still -o
nightsky.jpg --shutter
15000000 --gain 1
--awbgains 1,1 -
immediate
```

This will take a 15-second exposure as a high-quality JPG. You can read the docs further if you'd like to make further modifications, e.g. to the gain level of the capture: **magpi.cc/libcamera**.



The HQ Camera has a maximum exposure time of about ten minutes

Tip

The HQ Camera uses a C/CS-mount and M12 mounts for lenses, so you can use whatever lens you like, if it's compatible

ASTRO NAVIGATION

68 Creating star trail photos

70 Shooting with a telescope

Create star trail photos

Use long exposure and time-lapse techniques to create beautiful night shots

With the basic setup, you're able to create a couple of kinds of classic space shots – wonderful photos of the night sky, or stylish star trails. With a Python script, you can create the streaking star effect created by the Earth's rotation.

Hardware

USB drive

Taking a lot of big pictures can take up space really fast

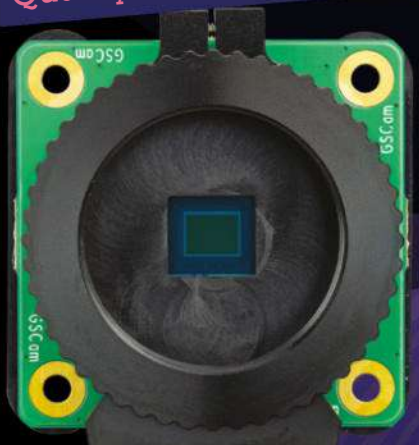
Screen (optional)

It's always a good idea to check what you're taking a photo of, although you can always use VNC if you don't have a spare screen

Raspberry Pi (any)

You can use any Raspberry Pi with a CSI port, which means most Raspberry Pi Zeros and any standard model. The newer the better, though

Raspberry Pi High Quality Camera and lens



See the previous page for tips on what lens to get – although the 16 mm one will be more than fine for this

Power bank

You'll want to do this outside, so a decent mobile battery is essential

Software

Some simple Python code that you can use to take a night full of shots. Put it on your USB drive

01 Setting up the software

For doing what is called a photo stack, we are going to be using the GNU Image Manipulation Program (IMP) from **gimp.org**. Once you've installed that, you'll also need a plug-in to create the stack. Grab this from **magpi.cc/starstack** and follow the instructions.

02 Get your photos

Grab the USB drive with your photos and put it into your PC. We suggest copying them to a new folder on your PC to work with – make sure you know where it lives! Once that's done, open IMP, go to File > Create > Startrail. Set 'Light Frames' to the folder you put all the photos you want to use in, and click OK.

> "You might be able to tailor the perfect shot"

03 More options

There are more options to play with, which you can find on the plug-in website, that can help you refine your photo. Take a look at them if you're not satisfied with your result, and you might be able to tailor the perfect shot.

star_trails.py

> Language: **Python**

```
001. #!/usr/bin/python3
002. import time
003.
004. from picamera2 import Picamera2
005.
006. picam2 = Picamera2()
007. picam2.configure("still")
008. picam2.start()
009.
010. # Give time for Aec and Awb to settle, before disabling
     them. Exposure time of 30 seconds
011. time.sleep(1)
012. picam2.set_controls({{"ExposureTime": 30000000,
     "AeEnable": False, "AwbEnable": False, "FrameRate": 1.0})
013. # And wait for those settings to take effect
014. time.sleep(1)
015.
016. start_time = time.time()
017. for i in range(1, 100): # 100 shots
018.     r = picam2.capture_request()
019.     r.save("main", f"image{i}.jpg")
020.     r.release()
021.     time.sleep(300) # Five minute delay for timelapse
022.
023. picam2.stop()
```

Tip

Depending on how big a USB drive you have attached, you can always increase the number of photos and exposure time (up to 60 seconds), and decrease the number of seconds between shots!

LOCATION, LOCATION, LOCATION

You'll obviously want to shoot at night, but where should you shoot from? The best place is somewhere away from a city or urban area where light pollution doesn't affect the view of the stars. The website **gostargazing.co.uk** has excellent recommendations for places to visit.

Shooting with a telescope

Get clear photos of distant objects using a telescope and smart software

To take amazing photos of celestial bodies, you need to be able to keep them in view, which means moving the camera. Luckily, with the right software, hardware, and a Raspberry Pi, you can take some amazing photos.

Hubble Pi

Student Santiago Rodriguez took a fairly simple approach of setting up a telescope for astrophotography with Raspberry Pi – he merely used an adapter for the eyepiece on a telescope so the HQ Camera could be plugged in.

With that done, he created a special UI called AstroCam that allows him to point the telescope at the right spot and then also control some of the camera settings, like the ones we've previously discussed. It's a lot cheaper as a result.

"Most good USB cameras for astrophotography start at about €200 and require a connected computer at all times. Hubble Pi can also work wirelessly, or as a standalone if needs be," he told us, when we interviewed him about it.

magpi.cc/hubblepi

PiFinder

A lot of modern telescopes can connect to computers to help with tracking distant objects. Richard (aka 'brickbots') had been doing a lot of manual checking with paper and then a commercial computer – he decided to build one based on Raspberry Pi.

It uses a mixture of GPS positional data and a visual feed from a HQ Camera to figure out exactly where the telescope is pointed, including a custom interface so it can be controlled out in the (literal) field.

"My hope is that other people will find this combination of functionality useful, will build their own PiFinder, and help the whole project improve by making suggestions and potentially contributing to the software. It's a pretty easy build with off-the-shelf parts and beginner-friendly soldering," Richard says on his website.

magpi.cc/pifinder

Astrophotography Autoguider

After spending many nights trying to get great shots from stacked photos – and too many not coming out very well – Joe Kutner decided to program his own autoguider that could keep his telescope pointing at the same celestial object.

> "Every step in the process had its challenges"

Making use of a touchscreen and Raspberry Pi, Joe created a ton of automated scripts that could be activated at the press of a button, and make sure his telescope was always pointing at the exact same object in the night sky.

"Every step in the process had its challenges," Joe told us a few years ago. "I would install one piece of software and then find out it wasn't compatible with some version of another piece of software I needed. When I finally got everything running, it wouldn't talk to my telescope until I installed yet another version of the software. There were dozens of these little paper-cuts but, in the end, it was worth working through them."

magpi.cc/autoguider

COSMIC EVENTS

Want to know when the best times for seeing specific objects is? The calendar from Sea and Sky has dates for all the major events in the night sky. Take a look here: **magpi.cc/astrocal**

Vizy

SPECS

COMPONENTS:
Raspberry Pi 4 (2, 4, or 8GB RAM), universal tripod mount, 32GB microSD, 25 W AC power supply, Vizy case (900 g; 10×15×10 cm)

CAMERA:
12-megapixel Sony image sensor (IMX477), wide-angle, 3.25 mm M12 lens (optional high-quality 8-50 mm C/CS zoom lens), electronically switchable IR-cut filter

I/O:
Digital I/O, analogue I/O, PWM, UART serial; high-current output (up to 1000 mA per channel), optional lighting accessory

▶ Pixy ▶ **vizycam.com** ▶ £280 / $259

This purpose-built AI camera combines Raspberry Pi with machine learning software. By **Lucy Hattersley**

Vizy is an interesting camera product that combines Raspberry Pi 4 with a Sony camera and lens. This is housed inside a custom plastic case with heatsinks, a fan, and a high-current I/O connection.

Once it's on your network, you can log in via a web browser pointing to **vizy.local**, and all the interaction takes place through this web-based interface. Vizy has a laudable aim of making it easy for students and makers to explore AI-powered machine learning projects.

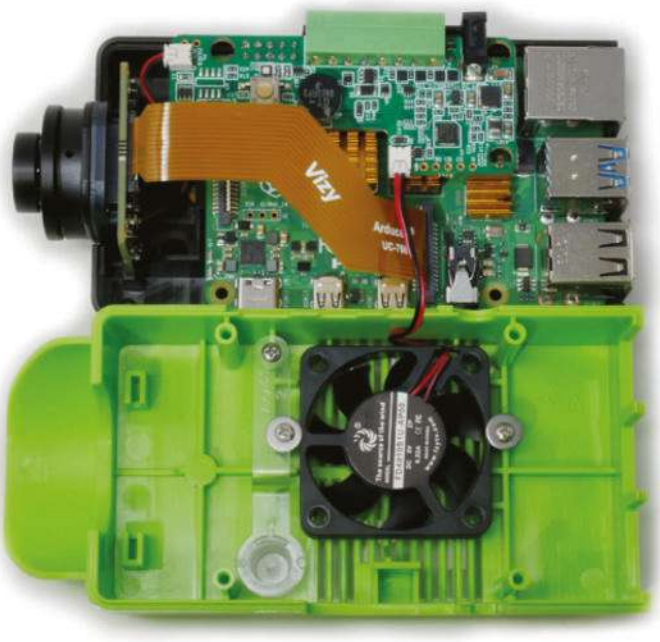
Vizy setup

Vizy comes self-assembled (you can choose between 2GB, 4GB, and 8GB Raspberry Pi inside). A microSD card is pre-installed and all you need to do is plug it in and press the button to turn it on.

While there's no fun in the build process, it is quick to get up and running. On first run, it uses Raspberry Pi's Wi-Fi module to broadcast a network, which you can join from another computer. You can also connect via an Ethernet cable, which was the approach we favoured.

Once connected, you set up Vizy to automatically connect to a local wireless network so it connects whenever you boot up. Then, you can investigate the built-in software, divided into a range of apps and examples.

Built-in apps include a Bird Feeder (identifies birds), MotionScope (captures the motion of objects), and Object Detector (detects and logs classes with text alerts sent via Signal).



▲ Inside Vizy is a separate daughterboard offering I/O connections and a fan to keep the unit cool

The examples are more simple, including OpenCV Edge Detection, Pet Companion, Pic Taker, TensorFlow Lite, and simple video capture. You can investigate the Python code for each example, to help you start to build your own apps. Detailed documentation at **docs.vizycam.com** includes a 'Getting started' guide, and information about accessories and applications, plus some simple API information. You can access the Python editor directly from the web interface, or log into the Shell and control Raspberry Pi directly.

These built-in programs ensure Vizy is a great platform for exploring machine learning possibilities. We particularly liked MotionScope, which captures the movement of an object (such as a

Vizy houses a Raspberry Pi 4 and camera inside a shell with everything you need to get started with machine learning image projects

bouncing ball) and provides detailed x, y, and z co-ordinates as graphs and a data dump. We can imagine this working incredibly well alongside calculus learning.

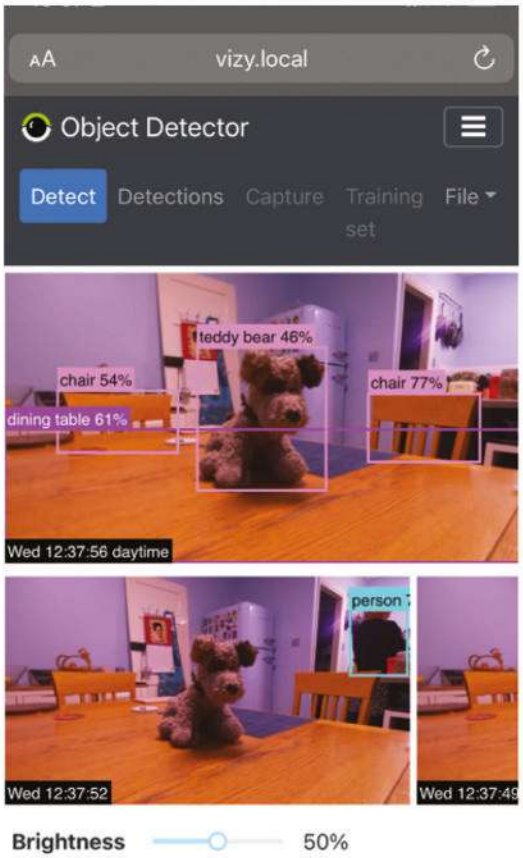
Meanwhile, the Object Detector and Bird Feeder can be used to analyse and respond to scenes,

> Vizy provides you with a great introduction to image recognition

creating alerts for specific items. We had a lot of fun tracking a pet cat and sending alarms when she was detected on a table.

The camera is the equivalent of a Raspberry Pi High Quality Camera, although we'd be happier if it used an official camera offering. We also found the 1.5m power lead to be a little short for some setups, mainly when we tried to move out into the garden for testing – although we note a Power over Ethernet Splitter (£15/$18, **magpi.cc/vizypoe**) that would work well in this setting. There is also an optional outdoor enclosure with IP66 rating and a 4G LTE cellular network adapter.

Vizy provides you with a great introduction to image recognition that does a lot of the heavy lifting, allowing you to focus on practical projects. We think this would be ideal in a classroom setting.



Here, we are accessing Vizy to set up a pet detector that sends alerts when a cat sits on the table (we are using a stuffed teddy bear as a test unit)

Verdict

An interesting camera that allows you to quickly investigate image recognition projects.

9/10

Meteor 10.1″ IPS Capacitive Touch Screen

SPECS

DISPLAY:
10.1″ IPS, 1200×800 pixels, 5-point multi-touch control

PORTS & CABLES:
HDMI port, 2 × micro-USB ports, HDMI cable, micro-HDMI adapter, 2 × micro-USB to USB cables

OTHER:
Raspberry Pi mounting point with screws, 2 × plastic stand legs

Verdict

The RGB marquee lighting is a bit gimmicky, but the picture quality is good and touch control works well. As a bonus, the screen can also be used with many other devices.

8/10

▶ Elecrow ▶ **magpi.cc/meteor10** ▶ £91 / $110

A versatile Raspberry Pi touchscreen with RGB animated lighting. By **Phil King**

▲ Whether you appreciate the RGB marquee lighting or not, the screen quality is very good

With a trail of glowing RGB LEDs around the edge and rear, the Meteor 10.1″ Touch Screen offers animated ambient lighting with 19 selectable effects. It can prove a little distracting, though, so luckily there's a button to turn it off.

Another button enables adjustment of the backlight for the screen, which is plenty bright enough at the higher settings. While 1200×800 pixels for a 10.1-inch display might seem on the low side (with a 143 PPI pixel density), in practice it looks detailed enough, even when playing video. There's no built-in speaker, though, so you'll need a separate one connected to your Raspberry Pi.

One plus point is the full-size HDMI socket, so you can easily plug in a Raspberry Pi using the supplied cable – with a micro-HDMI adapter for Raspberry Pi 4. There's a mounting point on the rear to secure Raspberry Pi to create an all-in-one unit, complete with a two-part plastic stand.

▲ You can mount a Raspberry Pi on the rear of the display using the supplied screws

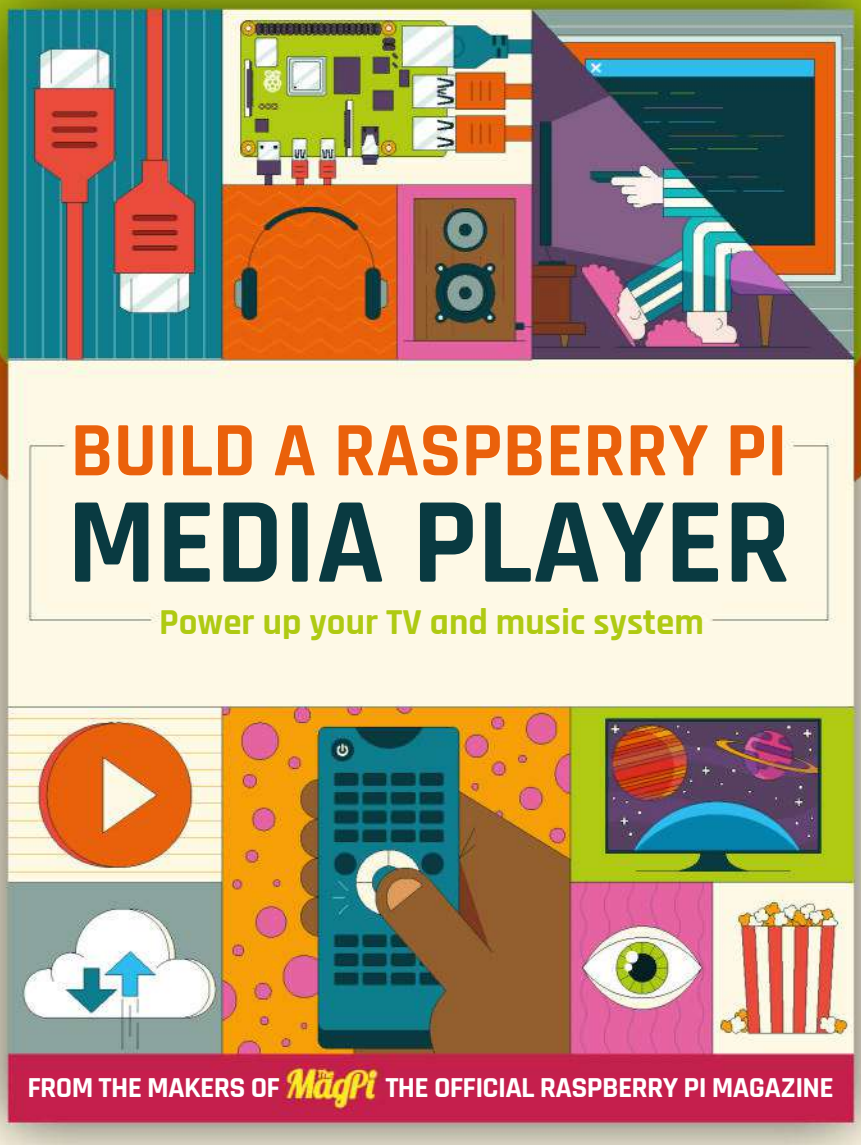
> There's a mounting point on the rear to secure Raspberry Pi to create an all-in-one unit

Touch control

Two micro-USB sockets need to be connected (via supplied cables) to Raspberry Pi's USB ports to supply power and enable touch control. An official PSU is advisable, otherwise the screen may be underpowered and keep switching off. In any case, it shows 'No signal' at times while Raspberry Pi OS is booting up, until the desktop appears.

We found it fiddly to tap icons and window controls, but this improved after changing the system default to 'Larger screens' to make them bigger. Pinch-zoom gesture control works well enough, although a bit of jiggery-pokery is needed in Raspberry Pi OS to enable right-clicking with a long press. Unless you want to attach a physical keyboard, you'll also need to install an on-screen virtual one such as Onboard.

Your FREE guide to making a smart TV


BUILD A RASPBERRY PI
MEDIA PLAYER
Power up your TV and music system
FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE

magpi.cc/mediaplayer

Alpakka

SPECS

LICENSES: Open-source firmware, Creative Commons hardware

INPUTS: 2 × gyro sensors, 13 × face buttons, 2 × sticks, 1 × scroll wheel, 6 × shoulder/grip buttons

OTHER FEATURES: Customisable, Raspberry Pi Pico-powered

▶ Input Labs ▶ **magpi.cc/alpakka** ▶ Free + cost of components

A DIY controller that can be used like a mouse or a gamepad with extra buttons. **Rob Zwetsloot** get to grips with it

DIY controllers are becoming more and more popular in the gaming space, especially when it comes to making accessible controllers for folks with disabilities, and 3D printing has really made it easy to roll your own input system.

Alpakka has taken all this and gone a little step further, creating a customisable controller base for (mostly) free and showing you how to put it together. We say mostly free because you do have to buy some extra components and get the PCB printed. However, even with a handful of components, you're not spending more than a tenner, especially if you have a Raspberry Pi Pico lying around to power it.

Ours came pre-assembled but the instructions are very clear, requiring you to flex only a few soldering skills to put it together.

Custom controls

Once everything is soldered, construction is fairly straightforward. Parts slot in and are tightened in place with a few screws, with the hardest part being the final sandwiching of the parts to finish the controller.

Build quality is quite dependant on how you print the controller, and also if you decide to customise it before sending the STL over to your printer. With all the electronic parts inside, it does have a satisfying weight and, thanks to the compact design, it doesn't feel loose and creaky like other 3D-printed handhelds and controllers we've used in the past.

Game on

As for how it plays – if you've ever used 3D-printed controls before, you'll know that they don't afford as much comfort as other controllers with softer and smoother plastic. The right stick isn't really supposed to act like a traditional right stick, so it sits there as an awkward cube that doesn't give the best look in first- or third-person games for aiming.

The left stick also feels a little clunky, and the triggers are hard to use as, thanks to the extra leverage you have on them, it's difficult to feel the

▲ It's powered by a Pico inside which allows the whole controller to work like a standard XInput controller

click on the button. They're also not analogue like on modern game consoles.

One of the reasons the controller is like this, though, is that you can also use it like a mouse thanks to advanced gyro functions – in fact, the scroll wheel on the controller is supposed to mimic one on a mouse. We found it a little awkward to use as we're not very used to it, but it definitely works better than older systems we've tried. It can also easily be turned off and on again, which is a bonus.

> With all the electronic parts inside, it does have a satisfying weight

For the price of putting this together at its most basic, it is very good though. It's great for more retro games, too, that don't require analogue controls and, due to its DIY nature, you could easily find more comfortable replacement buttons if you wanted to head down that route.

Rear grip buttons are also included, just in case you need more inputs

It's a very cool-looking controller with even cooler features packed inside

Verdict

We really like the idea of Alpakka, but if you want to use it as a main controller, you will need to start customising it yourself.

7/10

10 Amazing:

Monitors and displays

See what you're doing with these incredibly diverse add-ons

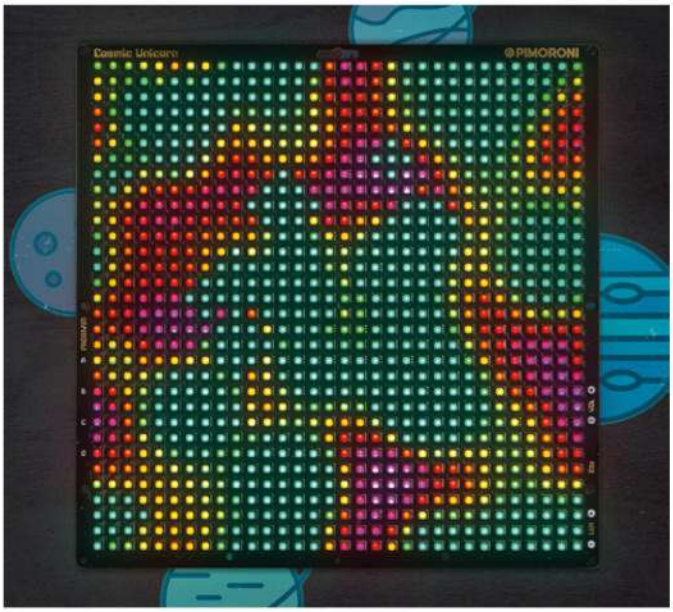
Using a Raspberry Pi without any sort of screen is fairly common. However, if you want to show something visually from Raspberry Pi, or even a Raspberry Pi Pico, you're spoilt for choice – not only in size but also in type. Here are ten of our favourites.

▲ HyperPixel 2.1 Round

HD circle

A tiny screen that fits neatly on top of a Raspberry Pi Zero. It's probably too big to be used as a watch, but no one will stop you trying.

magpi.cc/hyperpixel2r | £54 / $65



▲ Cosmic Unicorn

1024 colourful LEDs

The Unicorn line of LED matrices are a great alternative to a traditional display – and with so many LEDs, the Cosmic Unicorn might as well be a real display.

magpi.cc/cosmicunicorn | £80 / $97

▲ EPD Pico Development Kit

e-ink board

This add-on allows you to attach an e-ink display to Pico. It's on a cable, so Pico doesn't have to be mounted to it either.

magpi.cc/epdk | £25 / $30

▼ IPS wide touchscreen

Wide boy

If you need a display that is very wide or very long, this 7.9" touchscreen with a 16:5 aspect ratio is a great place to start.

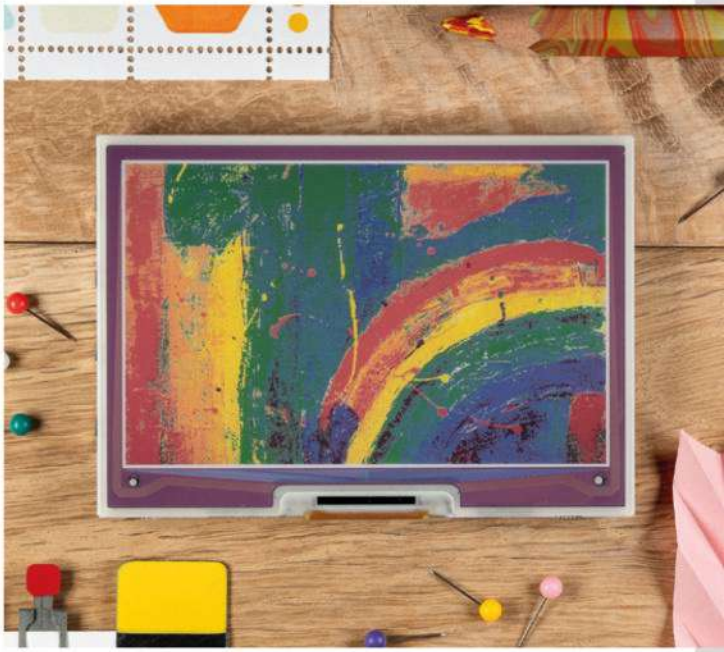
magpi.cc/ipswide | £85 / $103

HyperPixel 4.0

4-inch touchscreen

What we consider to be the standard small display for a Raspberry Pi, the HyperPixel 4.0 has been around for a while and is still top in its class.

magpi.cc/hyperpixel | £54 / $65



Inky Impression

E-paper add-on

This seven-colour display looks beautiful and uses very little power to keep itself going. Great for more arty projects.

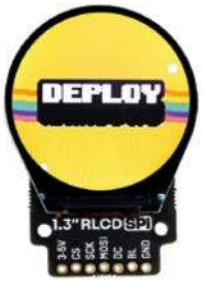
magpi.cc/inkyimpression | From £50 / $60

1.3″ OLED

Tiny display

This tiny screen fits snugly atop a Pico, with 64×128 pixels to play with. It even has a couple of buttons!

magpi.cc/oled13 | £11 / $11



Round LCD Breakout

SPI colour screen

This tiny round screen attaches to a Pimoroni Breakout Garden, slotting into one of the many ports in it.

magpi.cc/roundlcd | £25 / $30

Raspberry Pi Touch Display

Desktop ready

The official touchscreen slots into the DSI port on full-size Raspberry Pi and can be made quite compact if needed.

magpi.cc/touch | £69 / $83

GFX HAT

Old-school charm

We have a soft spot for the GFX HAT, giving you a very functional retro LCD display with six capacitive buttons to boot.

magpi.cc/gfxhat | £17 / $21

Learn Computer Science

Getting to grips with the lingua franca of how computers work is well worthwhile, recommends **Rosie Hattersley**

FutureLearn

AUTHOR

FutureLearn: IT & Computer Science Courses

Price:
£30 per month

magpi.cc/futurelearn

FutureLearn is an online learning portal with courses written by experts in their field drawn from a wide range of academic professional bodies. Unsurprisingly, you'll spot the Raspberry Pi Foundation and some very high-profile universities have contributed courses, with topics ranging from ethical hacking, artificial intelligence and robotics, to game development and data analysis, as well as courses focusing on particular software packages and computer programs. Clicking on the Overview tab takes you to a synopsis, expected prior knowledge, and what you'll learn. Most FutureLearn courses are included in the Unlimited recurring monthly subscription package. To make the most of this, clear your diary and tackle a couple of classes across different disciplines. Depending on your interests and chosen courses, you could accrue credits towards a degree or MBA.

Books

These bookshelf essentials contextualise computer science

CODE: THE HIDDEN LANGUAGE OF COMPUTER HARDWARE AND SOFTWARE

Author Charles Petzold looks at how computers and code impact our everyday lives and our interactions, from the way we describe things to our modes of communication.

▶ magpi.cc/codebook

THE SECOND MACHINE AGE: WORK, PROGRESS, AND PROSPERITY IN A TIME OF BRILLIANT TECHNOLOGIES

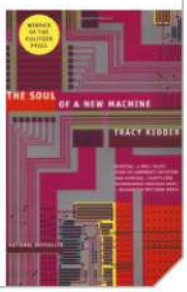
Unavoidable and sometimes awkward consequences as machines, robots, and artificial intelligence impact the way we work, influence society, and reward some at the expense of others.

▶ magpi.cc/2mabook

THE SOUL OF A NEW MACHINE

Tracy Kidder's seminal guide to the history of computer science is essential reading, introducing us to the industry's blow-hard business overachievers, grafters, and wingers with humorous insight.

▶ magpi.cc/soanm



CS50

AUTHOR
Harvard University

Price:
Free

magpi.cc/cs50

CS50x is Harvard's free, learn-at-your-own-pace introduction to computer science and the art of programming, and is the online equivalent to the most popular module available to campus attendees. The course is billed as ideal for learning about "thinking algorithmically" and solving problems efficiently. As well as encountering various programming languages such as C, Python, and JavaScript, and learning about SQL databases, you are expected to apply what you've learnt by tackling a range of data sets in order to demonstrate your understanding of cybersecurity, cryptography, gaming, and forensics. Score sufficiently highly on these tasks, and you will earn a prestigious Harvard University certificate. CS50 also provides a good grounding for related courses in web and games development, Python, and AI.

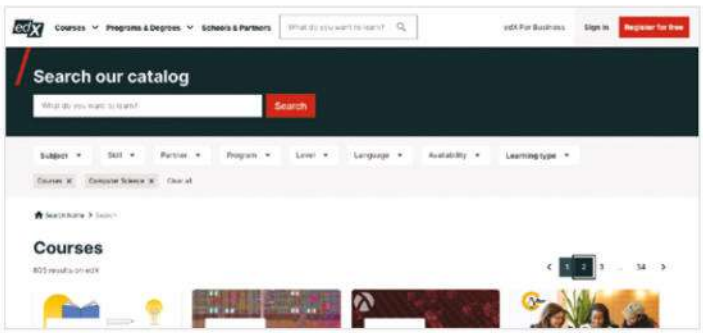
edX Computer Programming

AUTHOR
edX

Price:
Varies from free to $10,000 for full master's degree

edx.org

If TEDx is the place to go to hear insightful and inspiring talks about technology, edX is its online lecturer cousin. A helpful FAQs page makes it easier to choose between areas of study and select an appropriate academic (such as the University of Texas degree in Data Science) or general interest level, listed in the Executive section. Depending on the institution (e.g. MITx) and course selected, you can follow an academic CS track for free but forego personalised feedback and official certification, though paid-for upgrades are optional extras. Some courses are pitched at upskilling for vocational advancement, and others tailored to specific industries, which may be useful if you want to take advantage of funding for lifelong learning.



Websites

Places for advice and additional prompts

STACK OVERFLOW

If your code is broken, or you just can't think of a way to tackle something, appeal to fellow coders and get some expert advice.

▶ stackoverflow.com

JAVA T POINT

Great for help with a specific computing topic, or a detailed video overview of how to use a particular program, no matter how obscure it seems.

▶ javatpoint.com

ISAAC COMPUTER SCIENCE

The perfect site to head to if you're studying (or teaching) Computer Science at GCSE or above, as Isaac was developed together with UK exam boards in tandem with renowned computing companies such as Microsoft, Oracle, and Raspberry Pi.

▶ isaaccomputerscience.org

the

COMPUTERS THAT MADE BRITAIN

*"**The Computers That Made Britain** is one of the best things I've read this year. It's an incredible story of eccentrics and oddballs, geniuses and madmen, and one that will have you pining for a future that could have been. It's utterly astonishing!"*

- **Stuart Turton**, bestselling author and journalist

Buy online: **wfmag.cc/ctmb**

Available on amazon

odore

sinclair

OUT NOW

ACORN

ATARI

the

COMPUTERS

THAT MADE

BRITAIN

THE HOME COMPUTER REVOLUTION OF THE 1980s

Commodore

Apple Comput

TIM DANTON

sinclair

Apple Computer

Trevor Warren

An open-source advocate and community creator in Melbourne

> Name Trevor Warren | > Occupation Software engineer
> Community role Community leader | > URL hack2.live

We love High Altitude Ballooning (HAB) – we've been covering it in *The MagPi* basically as long as the magazine has been around. When Trevor emailed us about some launches he'd been doing in Australia, we were keen to talk to him, finding out he's set up several coding and making-related volunteer groups over the years.

"I have always been playing around with Linux and open-source growing up," Trevor tells us. "Linux was my gateway to learning more about computing and how computers worked… hardware was expensive and not really affordable in those days, the only way to dabble with electronics was to build everything yourself from scratch or use expensive commercial platforms which cost an arm and a leg. In many ways, Arduino democratized [the] learning of electronics, put electronics into the hands of makers. Raspberry Pi did to the SBC market what the Arduino did to the microcontroller market.

"I have been lucky to have had some really great experiences growing up. These experiences have given me the skills, confidence, ability to build and grow community organizations. I consider myself privileged to be able to set up and run these volunteering groups (Melbourne Raspberry Pi Makers Group, CoderDojo Altona North, EMDRC High Altitude Balloon Group) here in Melbourne, Australia."

▼ Preparng for launch with a HAB filled just right

When did you first learn about Raspberry Pi?

I read about Raspberry Pi in a magazine, and ended up purchasing my first Raspberry Pi from a local electronics store. I was quite comfortable with Linux, so getting started with Raspberry Pi wasn't such a big deal for me. However, I never really found much application for my Raspberry Pi 1 Model A. It would be a few more years before I realized the importance of Raspberry Pi, and its power as a learning platform for makers and tinkerers.

Successful payload retrieval

Trevor also helps run CoderDojos

What drew you to organising community events?
It all started when I mentioned the Raspberry Pi to a few of my colleagues at work – some of them expressed interest and wanted to learn more about it, but didn't know where to start.

> We shared our experiences with Raspberry Pi, helped each other out

So, we rented out a room at the local library on a Saturday seven years ago, and got together as the Melbourne Raspberry Pi Users Group. We shared our experiences with Raspberry Pi, helped each other out with the issues we were having, while helping the newbies set up their own Raspberry Pis – connecting it to the network.

I didn't know, at that time, that the Melbourne Raspberry Pi Makers Group was going to be such an important part of my life. I am honoured to be able to serve the community of makers here in Melbourne, Australia. It's a privilege to work with makers here, give them an opportunity to showcase the work they are doing, and creating a platform that brings makers with diverse interests together in person and online. We have been meeting in person at the Docklands Makerspace here in Melbourne (thanks to the Melbourne City Council for their support), and stay in touch online through Discord.

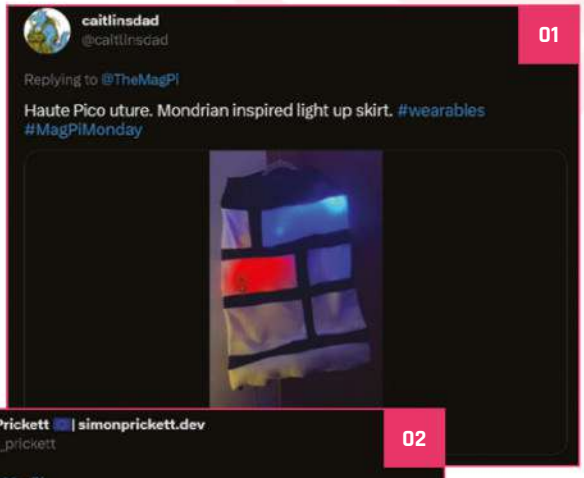
What standout memories do you have of previous events?
Every event is a memorable event for me. Things are very different now, as compared to when I started off with a group of four makers at the rented room in the council library. We now have a group of over 1000+ makers within the Melbourne Raspberry Pi Makers Groups on Meetup, and over 300+ makers on Discord. 2020 was hard on everyone, hard on the makers and their families. But we used it to bring us together, to support each other, and stay in touch with each other through Discord. Discord has been such a lifesaver for all of us. We now use Discord extensively and stay in touch between the monthly meetups using Discord.

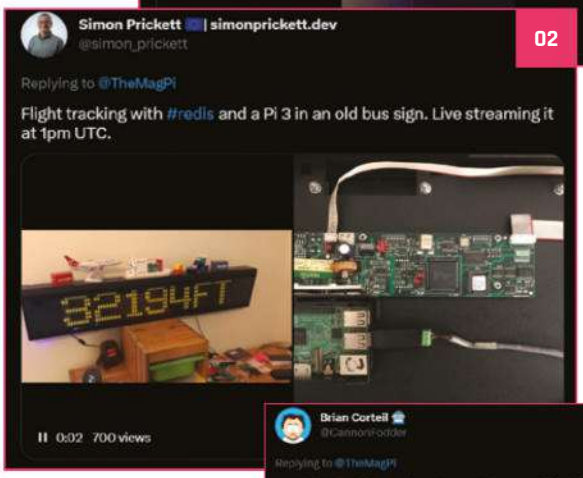
MagPi Monday

Amazing projects direct from our Twitter!

Every Monday we ask the question: have you made something with a Raspberry Pi over the weekend? Every Monday, our followers send us amazing photos and videos of the things they've made.

Here's a selection of some of the awesome things we got sent this month – and remember to follow along at the hashtag #MagPiMonday!




Simon Prickett | simonprickett.dev
@simon_prickett
02
Replying to @TheMagPi
Flight tracking with #redis and a Pi 3 in an old bus sign. Live streaming it at 1pm UTC.
32194FT
0:02 700 views

Brian Corteil
@CannonFodder
03
Replying to @TheMagPi
A quick make this week, @pimoroni plasma stick with a Pico W onboard controlled infinity mirror, there are 45 Neopixels in it. #MagPiMonday on a Tuesday :-)

01. The perfect gear for going clubbing with. Your friends will always find you.
02. We love these kind of upcycling projects – and it's a fun use of an old display!
03. Infinity mirrors will always be fascinating to us.
04. As well as looking cool, this is also a very neat visualisation of how analogue sticks work.
05. Always nice to give plants a fancy home.
06. This arcade stick looks great and, despite its warnings, will not electrocute you.
07. A simple yet very useful project – we need more little info screens in our life.
08. This soil moisture sensor project keeps getting better.
09. Another happy robot-building customer.

05

Kevin McAleer 🤖 Robot Maker
@kevsmac

Replying to @TheMagPi

Happy #MagPiMonday (ahem).
This weekend I created a self-watering plant system using the @pimoroni Grow Hat.

I created these cute geometric pots to give the plants a home.

Write-up here: kevsrobots.com/blog/self-wate...

#MonthOfMaking

04

Dr Footleg - Roboteer @drfootleg@fosstodon.org
@drfootleg

Replying to @TheMagPi

Started working on a robot remote controller using these 3 axis joysticks. First up, I needed a way to visualise joystick position from a distance so I can test the range once I get remote communication working.
#MonthOfMaking #MagPiMonday

Dr Footleg - Roboteer @drfootleg@fosstodon.org @drfootleg · Mar 10
Project progress, with an ESP-32 and the 3 axis analogue joystick from @pimoroni
Show this thread

06

Roland Schulz r.schulz.maker@mastodon.social
@r_schulz_maker

Replying to @TheMagPi

Hello #MagPiMonday on a cloudy Tuesday 😊👍, here is my unbelievable photo ! My Picade-Travelling-Wave-Tube-Pi 4 is ready for action, after wiring, printing and setup 🤓👍😎!

08

Pater Practicus
@PaterPracticus

Thanks for all the advice on sleep modes for @Raspberry Pi #Pico in response to yesterday's #MagPiMonday post. Brief mention at the end of my new soil moisture monitor video: youtu.be/E7DYY6i6u0M but definitely needs further investigation. @TheMagPi

07

Kiting Free
@KitingFree

Replying to @TheMagPi

I've been working this weekend on turning a pico-based @pimoroni badger into an all-purpose information center for my kitchen (not yet ready!)

09

Roland Schulz r.schulz.maker@mastodon.social
@r_schulz_maker

Replying to @TheMagPi

#MagPiMonday I built a litte #RasberryPi Pico robot with a industrial limit-switch like @biglesp described in his article at @tomshardware website published May 2021.

Events in pictures

Raspberry Jams are back, and we have the photos to prove it

CODERDOJO BAARLE RASPBERRY PI BIRTHDAY PARTY

What better way to celebrate the birthday of Raspberry Pi than coding?

The Astro Pi Kit gets a good workout here

MELBOURNE RASPBERRY PI MAKERS GROUP MEETUP

What does the robot see?

The meetups occur at the Docklands Makerspace & Library

Crowdfund **this!**

Raspberry Pi projects you can crowdfund this month

UHF Reader

"A Rain RFID module built into both models can scan up to 50 tags per second within a 1.5-metres range. The Ultra-High Frequency (UHF) band, which ranges from 840MHz to 960MHz, is where this device operates. One reader may interact with numerous tags at once, fast, and within more than a metre of one another because of its wider read range than HF NFC scanners."

▶ **kck.st/3SIgNYx**

Raspled Digital Signage

Raspberry Pi is used in digital signage a lot – which is why you'll sometimes see people post bemused pictures of a sign in an airport showing off its Raspberry Pi innards. Apparently, a lot make use of monthly fees for a very simple operation, which maker Angel Rijo says won't happen here.

▶ **kck.st/3ZFYvJr**

Best of the rest!

Other amazing things we saw this month

N64 Digital HDMI converter

Using older consoles with an analogue output on modern TVs is fraught with various quality issues. However, this amazing project uses a Pico to convert the signal better than some solutions.

▶ **magpi.cc/n64hdmi**

Solar rover

If there's one way to explore the Australian outback without being attacked by natural horrors, it's through the safety of a video feed attached to a remote-controlled robot.

▶ **magpi.cc/solarrover**

Train logger

User iHoller913 was interested to know if there was any pattern to the trains going by their home, resulting in this professional-looking device.

▶ **magpi.cc/trainlogger**

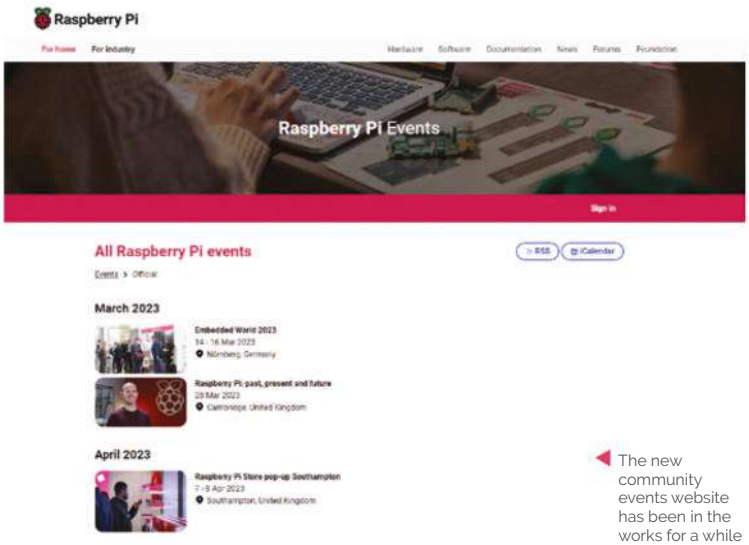
Your Letters

Raspberry Jams

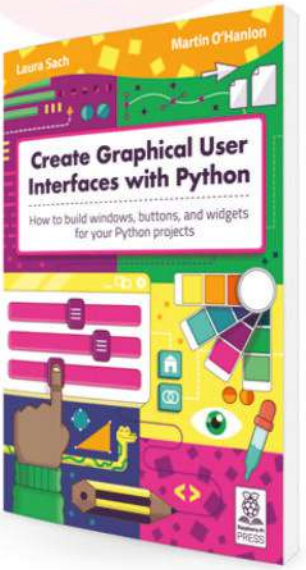
I read about CamJam coming back and was wondering if the calendar with all the Jams would be too? It helped introduce me to a few events in my area a few years back, and was hoping I might find some more if it came back.

Mel via Twitter

This issue sees the triumphant return of our community events pages – you can check them out over the page – and they're backed up by the new events site that you can find at **magpi.cc/events**. Hopefully, the info will help you find more local events and friends!



The new community events website has been in the works for a while


Laura Sach
Martin O'Hanlon
Create Graphical User Interfaces with Python
How to build windows, buttons, and widgets for your Python projects

Learn about making interfaces with Python with our excellent book

Future article request

I wonder if, in the future, you could print articles on the following:

gpio readall – the installation of and usage of as a means of checking the status of inputs and outputs.

Python scripts on a Raspberry Pi – in particular, the means of running a script that gets data from an API and displays the data on PyQt5 GUI, then, after a time delay, getting fresh data from the API and displaying this new data.

Marian via email

It's not quite PyQt5, but our book *Create Graphical User Interfaces with Python* (**magpi.cc/pythongui**) does cover this kind of program inside. A lot of the parts that reference and access API data are the same, you'd just need to rework the outputs to work with PyQt5. We'll definitely keep it in mind for future articles on Python programming, though.

As for finding out the status of GPIO – it's not something we've covered, but it is part of WiringPi. Definitely something we will talk about in the future.

Space is now

In *The MagPi* #125, the Next Month page (97) featured astrophotography. However, in *The MagPi* #126, the main focus was on the new Camera Module, with little to nothing pertaining to astrophotography. Will this be a feature of a future magazine?

Andy via email

As you've probably noticed by now, this issue has the astrophotography feature we were originally planning for #126. Sometimes plans change behind the scenes over the month between issues, although we do prefer to keep our plans consistent. We think the feature this issue will have been worth the wait.

▲ Check out our astrophotography feature and make amazing time-lapse photos like this

Contact us!

- Twitter **@TheMagPi**
- Facebook **magpi.cc/facebook**
- Email **magpi@raspberrypi.com**
- Online **forums.raspberrypi.com**

USA SPECIAL!
6 ISSUES
FOR $42

FREE
RASPBERRY PI PICO W

Subscribe online:
magpi.cc/subscribe

Price is charged at £35 sterling. The dollar price will depend on the exchange rate. Six issues and free Pico W for £35 is also available in Canada and Europe. Subscription is for the next six issues and does not renew automatically. This is a limited offer. Offer subject to change or withdrawal at any time.

Community Events Calendar

Find out what community-organised Raspberry Pi-themed events are happening near you...

01. Melbourne Raspberry Pi Makers Group Meetup

- Sunday 2 April
- Library at the Dock, Melbourne, Australia
- magpi.cc/mrpmgm128

This meetup is open to everyone with an interest in electronics, robotics, home automation, 3D printing, laser cutting, amateur radio, high altitude balloons, space tech, etc. Makers are invited to bring along their projects and project ideas, and come connect with other makers. Get your questions answered, show off the work you are doing, and get support to resolve nagging issues.

02. Cornwall Tech Jam

- Saturday 15 April
- Fraddon Village Hall, Saint Columb, UK
- magpi.cc/ctj128

Cornwall Tech Jams are run by volunteers working in IT and education throughout Cornwall. They are supported by Software Cornwall, its members and other local businesses. Our volunteers give their own time and expertise to plan each Cornwall Tech Jam and to the regular maintenance of all our equipment.

03. Pi and More 13

- Saturday 22 April
- Trier University of Applied Science, Trier, Germany
- magpi.cc/pam13

At Pi and More, beginners and experts meet in a relaxed atmosphere for talks, workshops, and projects focusing on Raspberry Pi, embedded systems, and microcontrollers.

04. Cambridge Raspberry Jam

- Saturday 22 April
- Makespace Cambridge, Cambridge, UK
- magpi.cc/camjam128

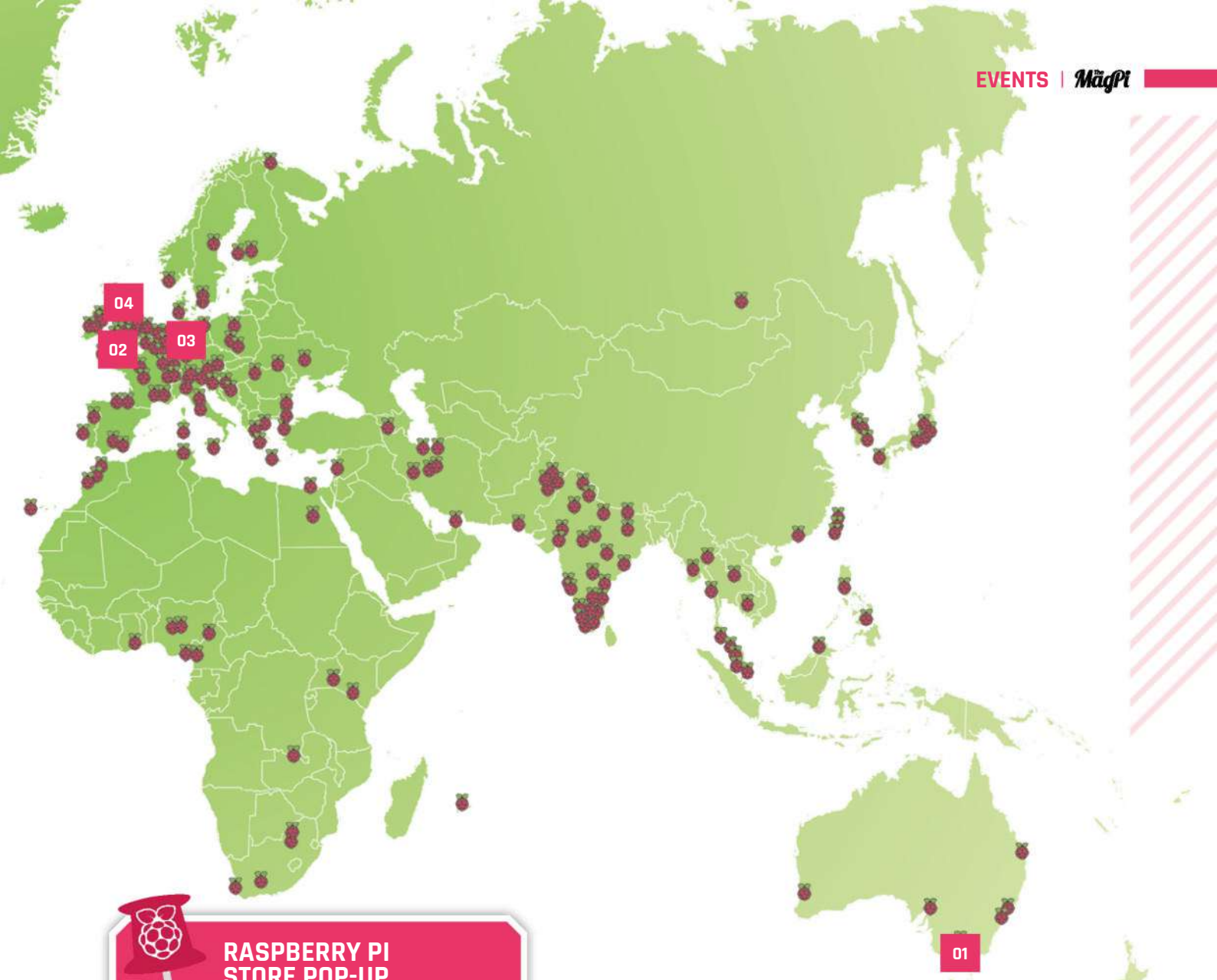
CamJam is back! Cambridge Raspberry Jam (known as CamJam) is an event and meetup for those interested in the Raspberry Pi family of computing devices and other technologies which encourage making and education.

There will be a series of talks, organised workshops for both the Raspberry Pi and Pico, plus a 'show and tell' area for people to show off their projects.

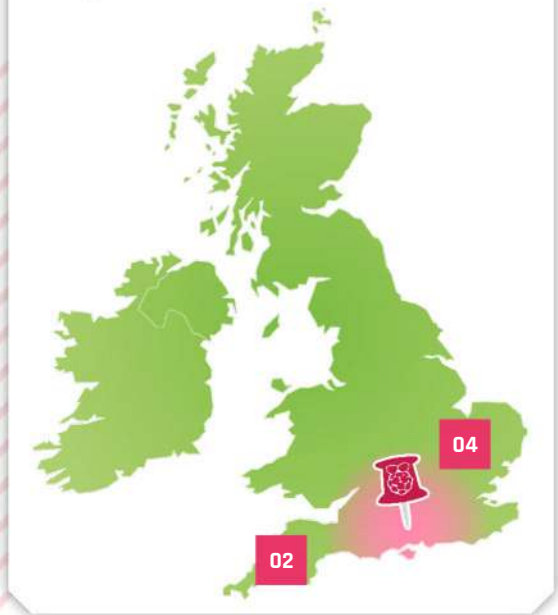
FULL CALENDAR

Get a full list of upcoming community events here:

magpi.cc/events



RASPBERRY PI STORE POP-UP



Where can you find Raspberry Pi next?

- Next location Raspberry Pi Store pop-up Southampton
- Where West Quay Shopping Centre, Southampton, UK
- When Friday 7 April and Saturday 8 April

At this Raspberry Pi Store pop-up in Southampton, you can experience and buy Raspberry Pi products. Explore some of the things you can do with a Raspberry Pi, discover our accessories and books, and get your hands on limited edition exclusives.

If you're lucky, you might spot Features Ed. Rob!

magpi.cc/popup2023soton

HackSpace

TECHNOLOGY IN YOUR HANDS

THE MAGAZINE FOR THE MODERN MAKER

SUBSCRIBE AND SAVE UP TO 35% on the cover price

ISSUE **#65**

OUT NOW

hsmag.cc

Available on the App Store

GET IT ON Google Play

WIN A RETERMINAL

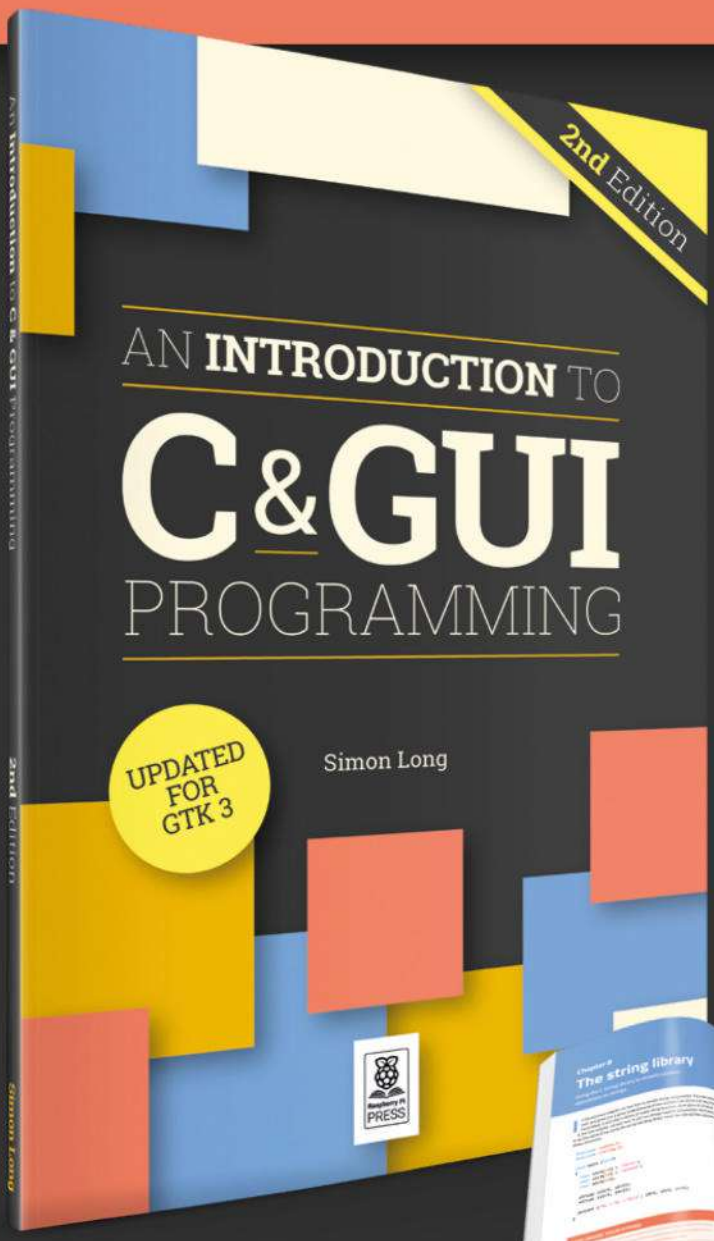
A product on our ever-growing list of excellent Raspberry Pi Compute Module 4 products, reTerminal is a hardened production terminal for factories, or other industrial-level monitoring. It's also modular, allowing you to expand its functions.

Head here to enter: magpi.cc/win | Learn more: magpi.cc/reterminal

Terms & Conditions

Competition opens on **29 March** and closes on **27 April**. Prize is offered to participants worldwide aged 13 or over, except employees of Raspberry Pi Ltd, the prize supplier, their families, or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram, Facebook, Twitter or any other companies used to promote the service.

AN **INTRODUCTION** TO C&GUI PROGRAMMING



All you need to know to write simple programs in C and start creating GUIs

Inside:

- Create simple command-line C programs
- Control flow with conditions and loops
- Handle variables, strings, and files
- Design graphical user interface applications in C
- Handle user input with buttons and menus
- Use advanced UI features such as data stores and dialogs

£10 with FREE worldwide delivery

Buy online: magpi.cc/cgui

HOME AUTOMATION WITH RASPBERRY PI

Let's wire up your house

THE MAGPI **#129**
ON SALE **27 APRIL**

Plus!

A LEGO brick that plays Doom

ChatGPT with Raspberry Pi

Learn Sense HAT

DON'T MISS OUT!
magpi.cc/subscribe

TWITTER	@TheMagPi
FACEBOOK	fb.com/MagPiMagazine
EMAIL	magpi@raspberrypi.com

EDITORIAL

Editor
Lucy Hattersley
lucy@raspberrypi.com

Features Editor
Rob Zwetsloot
rob@raspberrypi.com

Sub Editor
Nicola King

ADVERTISING

Charlotte Milligan
charlotte.milligan@raspberrypi.com
+44 (0)7725 368887

DESIGN

criticalmedia.co.uk

Head of Design
Lee Allen

Designers
Olivia Mitchell, Sam Ribbits

Illustrator
Sam Alder

CONTRIBUTORS

David Crookes, Ben Everard, Rosemary Hattersley, Gordon Hollingworth, Nicola King, Phil King, KG Orphanides, Eben Upton, Stewart Watkiss

PUBLISHING

Publishing Director
Brian Jepson
brian.jepson@raspberrypi.com

Director of Communications
Liz Upton

CEO
Eben Upton

DISTRIBUTION

Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

SUBSCRIPTIONS

Unit 6 The Enterprise Centre
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE
+44 (0)1293 312193
magpi.cc/subscribe
magpi@subscriptionhelpline.co.uk

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

The MagPi magazine is published by Raspberry Pi Ltd, Maurice Wilkes Building, St. John's Innovation Park, Cowley Road, Cambridge, CB4 0DS. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products, or services referred to or advertised in the magazine. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0).

ISSN: 2051-9982.

Let's get together

Raspberry Pi events will bring our community together again. **By Lucy Hattersley**

Raspberry Pi is as much a community as a computer, and I am utterly overjoyed to see the return of the brilliant Events pages to this magazine (page 92).

Rob Zwetsloot runs our community section and will be a familiar face to most of you. Rob loves Raspberry Pi and our community and is – I'm sure – as happy as I am to see real-world events taking place again.

Raspberry Pi itself is deeply connected with industry professionals and enthusiasts, and a new Events section can also be found on Raspberry Pi's website (**magpi.cc/events**).

Here, you'll find Raspberry Pi at industry trade shows and hobbyist events. Kicking off with Embedded World 2023, where you can meet Raspberry Pi makers. Eben Upton will be talking about "Raspberry Pi: past, present, and future" at the University of Cambridge (**magpi.cc/camfest**), and we will be at the Write the Docs festival in Portland, OR. You can also visit Raspberry Pi at its store in Cambridge, or a pop-up store coming soon to Liverpool.

And this is just the beginning of official Raspberry Pi events. There are also Raspberry Pi community events: Tech Jams, meetups, and maker space days.

Team huddle

The Raspberry Pi community is a vibrant place, where industrial makers, enthusiastic programmers, educators, and hobbyists all gather to discuss projects and share ideas. This diverse group of people is brought together by a love of computing and the idea that 'making makes the world a better place.'

> This diverse group of people is brought together by a love of computing and the idea that making makes the world a better place

This love manifests itself via Raspberry Pi, because it's the best low-cost, single-board computer, around; but it'd probably exist anyway. I've tried lots of different computers, and Raspberry Pi is the best – mostly because it's so versatile. This small, low-cost computer is used for a wide range of applications, from building robots to running servers; playing retro games and automating your home. And, of course, coding and computing for its own sake. I'm happy to mess around with some abstract concepts in Visual Studio Code.

Perhaps the most important part of the Raspberry Pi community is the spirit of collaboration that brings everyone together. Whether you're a total beginner or an experienced professional, everyone is welcome, and everyone is encouraged to share their knowledge. This supportive atmosphere is reflected in the many online forums, blogs, and social media groups that are dedicated to the Raspberry Pi and the projects that can be built with it.

We're currently in the midst of our #MonthOfMaking event, but you can share whatever you've made with *The MagPi* team using #MagPiMonday. Your build could make it into these hallowed pages.

AUTHOR

Lucy Hattersley

Lucy is editor of *The MagPi* and makes the magazine with Rob, Nicola, Olivia, and some extremely talented people. It's very much a team effort.

magpi.cc/lucyhattersley

HiPi.io

HIGHPI PRO

The new case from the HiPi.io team

- Rapid tool-free assembly and disassembly
- Large internal volume for HATs
- Compatible with Pi 2/3/4
- Multiple lid options
- Passive & Active Cooling options
- Secure microSD card cover
- VESA mount support
- Molding-configurable output ports customizable for volume orders
- Printed logo for your branding

Available at these great Pi stores:

PiShop.us PiShop.ca PiShop.mx ThePiHut Pi-Shop.ch Welectron.

Contact your favorite Pi store if it's not listed here

HiPi.io

PiKVM

Manage your servers or workstations remotely

A **cost-effective** solution for data-centers, IT departments or remote machines!

PiKVM HAT
for DIY and custom projects

Pre-Assembled version

- Real-time clock with rechargeable super capacitor
- OLED Display
- Bootable virtual CD-ROM & flash drive
- Serial console
- Open-source API & integration
- Open-source software

Available at the main Raspberry Pi resellers

core ELECTRONICS · PiShop.ca · RPishop.cz · Welectron. · elektorstore · PIMORONI

Pi-Shop.ch · ThePiHut · PiShop.us · CloudFree · PiShop.mx

Reseller suggestions and inquiries:
wholesale@hipi.io